CS Exit Ethiopia

Questions & Worked Answers

Computer Science Exit Examination

3,485 complete questions and 24 source variants

16 selected courses | 18 November 2026

Selected courses

01. Computer Programming
02. Fundamental of Database Systems
03. Advanced Database Systems
04. Object Oriented Programming
05. Computer Organization and Architecture
06. Data Communication and Computer Networking
07. Data Structure and Algorithms
08. Web Programming
09. Operating System
10. Software Engineering
11. Design and Analysis of Algorithms
12. Introduction to Artificial Intelligence
13. Computer Security
14. Network and System Administration
15. Automata and Complexity Theory
16. Compiler Design

https://csexitethiopia.com

abelasnake2003@gmail.com | Telegram: @Abechu7

Course contents

Each course begins on a new page. PDF bookmarks provide question, study-note section and revision-topic navigation.

Computer Programming	3
Fundamental of Database Systems	248
Advanced Database Systems	548
Object Oriented Programming	886
Computer Organization and Architecture	1324
Data Communication and Computer Networking	1604
Data Structure and Algorithms	2108
Web Programming	2485
Operating System	2878
Software Engineering	3203
Design and Analysis of Algorithms	3602
Introduction to Artificial Intelligence	3788
Computer Security	4098
Network and System Administration	4337
Automata and Complexity Theory	4501
Compiler Design	4693

Reading the answers

Every question includes its answer, explanation, choice-by-choice contrasts and available evidence. Primary note proof quotes the study notes. Related reading and module passages are labelled separately; a module reference is not presented as primary note proof.

Computer Programming

Questions & Worked Answers

177 questions + 1 source variants

Question 1 | Q00017

In the conventional historical classification of programming-language generations, which generation is associated with declarative logic, constraints, AI and expert systems?

A. Fifth generation languages

B. Second generation languages

C. Fourth generation languages

D. Third generation languages

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: A. Fifth generation languages

The question uses the historical generation taxonomy, rather than asking which languages can implement AI. The note associates fifth-generation languages with logic, constraints and expert systems. Second generation refers to assembly, third to general high-level languages, and fourth to query languages and application generators. Those other languages can be used within AI applications, but that does not change their generation category.

Why each choice is right or wrong

A - Correct: Fifth generation is the historical logic/constraints and expert-system category.

B - Incorrect: Second generation denotes symbolic assembly.

C - Incorrect: Fourth generation denotes query languages and application generators.

D - Incorrect: Third generation includes general high-level languages such as C++ and Java.

Primary note proof

Study-note section CP03

> Machine instructions are conventionally first generation; symbolic assembly is second; general high-level languages such as C++ and Java are third. Database query and application generators such as SQL are conventionally fourth generation. Fifth generation is a loose historical category associated with declarative logic, constraints and AI/expert systems. Generations do not determine whether an implementation uses an interpreter.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

MOE Computer Science Exit Exam · 2016 · Question 17

Question 2 | Q00027

Which one of the following is correct ordering in language processing system?

A. Source program ->Preprocessor -> Assembler -> Compiler Linker/Loader -> target machine code

B. Source program -> Preprocessor ->Linker/Loader -> Compiler /Assembler ->target machine code

C. Source program Compiler -> Assembler -> Preprocessor > Linker/Loader -> target machine code

D. Source program _> Preprocessor -> Compiler -> Assembler -> Linker/Loader -> target machine code

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Preprocessor → compiler → assembler → linker/loader.

CP04 explicitly places preprocessing before compilation proper and compilation before assembly. The assembler produces object files, which are linked into an executable and then loaded. D preserves that order, although it compresses linking/loading into one label and describes the final result loosely as target machine code.

Why each choice is right or wrong

A - Incorrect: Places the assembler before the compiler, reversing their stated order in CP04.

B - Incorrect: Attempts linking/loading before compilation and assembly have produced the required object files.

C - Incorrect: Places preprocessing after compilation and assembly, contrary to the conventional toolchain order.

D - Correct: Preserves preprocessing, compilation, assembly, linking and loading in the required sequence.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 27

Question 3 | Q00042

What does this C++ fragment print? Assume cout is available.

```
int p,q,n;
p=q=n=8;
p+=n;
p/=q;
cout<<p;
```

A. 16

B. 2

C. 24

D. 1

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: B. 2

All three integers initially hold 8. The compound addition changes p to 16 without changing q or n. The compound division then divides the updated p by q, giving 16/8=2. Only the final value is inserted into cout. Both compound operators are single tokens and follow ordinary integer update rules.

Why each choice is right or wrong

A - Incorrect: 16 is the intermediate value before division.

B - Correct: After addition and division, p is 2.

C - Incorrect: 24 would require another addition that the fragment does not perform.

D - Incorrect: 1 would result from dividing the original 8 by 8 and ignoring the addition.

Primary note proof

Study-note section CP08

> x+=y, x-=y, x*=y, x/=y and x%=y update x using the operation and conversion to its type. For simple variables, trace them as x=x op y. Starting p=q=n=8, p+=n gives 16 and p/=q gives 2. Starting x=10, x+=5 then x/=2 gives 7. x=x+x doubles; it does not generally add one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 42

Question 4 | Q00049

Look at the following fragment code :
int a = 6;
int b = 5;
while(a>b)
{
Cout<<"computer science program"<<endl;
}
Which algorithm property is not satisfied by the above C++ code algorithm?

A. Sequential

B. Feasibility

C. Finiteness

D. Definiteness

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer C: Finiteness

With a=6 and b=5, the while condition remains true because neither variable changes. Thus the intended algorithm never terminates and violates finiteness, explicitly required on PDF p.2. The steps are definite and executable; sequentiality is not the termination property. Interpret Cout as the intended cout. Module support: PDF p. 2, PDF p. 11.

Why each choice is right or wrong

A - Incorrect: "Sequential" does not match the rule or result derived here. With a=6 and b=5, the while condition remains true because neither variable changes. Thus the intended algorithm never terminates and violates finiteness, explicitly required on PDF p.2. The steps are definite and executable; sequentiality is not the termination property. Interpret Cout as the intended cout.

B - Incorrect: "Feasibility" does not match the rule or result derived here. With a=6 and b=5, the while condition remains true because neither variable changes. Thus the intended algorithm never terminates and violates finiteness, explicitly required on PDF p.2. The steps are definite and executable; sequentiality is not the termination property. Interpret Cout as the intended cout.

C - Correct: "Finiteness" matches the derived result. With a=6 and b=5, the while condition remains true because neither variable changes. Thus the intended algorithm never terminates and violates finiteness, explicitly required on PDF p.2. The steps are definite and executable; sequentiality is not the termination property. Interpret Cout as the intended cout.

D - Incorrect: "Definiteness" does not match the rule or result derived here. With a=6 and b=5, the while condition remains true because neither variable changes. Thus the intended algorithm never terminates and violates finiteness, explicitly required on PDF p.2. The steps are definite and executable; sequentiality is not the termination property. Interpret Cout as the intended cout.

Primary note proof

Study-note section CP02

An algorithm gives precise, executable steps and terminates for every legal input.

Study-note section CP16

A while(a>b) with fixed a=6,b=5 never changes its true condition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 19-26

Algorithm is an ordered set of unambiguous, executable steps, defining a terminating process. Important Properties of Algorithms - Correct: always returns the desired output for all legal instances of the problem. - Unambiguous: Each step must be clearly defined, having only one interpretation. - Precise: simple but efficient - Efficient: can be measured in terms of Time and Space - Finiteness: the execution of the algorithm must terminate in a finite amount of time for each possible set of input data/must have an end.

Computer Programming module | PDF page 11 | lines 5-13

Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 49

Question 5 | Q00051

In the conventional C++ build process, which program resolves external symbols and combines object files and libraries into an executable before it is loaded into memory?

A. Editor

B. Debugger

C. Linker

D. Loader

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: C. Linker

Combining object files and resolving their external references is the linker's role in the conventional toolchain. This occurs after object-file generation and before the executable is loaded for execution. Loading has a separate responsibility: placing the executable and required components in memory and starting it. The question asks about the build operation, so an editor, debugger or loader does not satisfy it.

Why each choice is right or wrong

A - Incorrect: An editor changes source text.

B - Incorrect: A debugger investigates program execution and defects.

C - Correct: The linker combines objects/libraries and resolves external references.

D - Incorrect: The loader places an already linked executable into memory.

Primary note proof

Study-note section CP04

> The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 51

Question 6 | Q00052

Machine language is grouped under_____ generation of programming language.

A. 2nd generation language

B. 3rd generation language

C. 4th generation language

D. 1st generation language

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Machine language is first generation.

CP03 calls machine instructions first generation and symbolic assembly second generation. The question asks specifically about machine language, so the first-generation option follows directly. Third and fourth generations refer to higher-level languages and query/application generators, respectively; they do not classify native machine instructions.

Why each choice is right or wrong

A - Incorrect: Second generation identifies symbolic assembly, whose instructions still require assembly into machine code.

B - Incorrect: Third generation identifies general high-level languages such as C++ and Java.

C - Incorrect: Fourth generation identifies query languages and application generators, rather than machine instructions.

D - Correct: First generation is the note's explicit conventional classification for machine instructions.

Primary note proof

Study-note section CP03

> Machine instructions are conventionally first generation; symbolic assembly is second; general high-level languages such as C++ and Java are third. Database query and application generators such as SQL are conventionally fourth generation. Fifth generation is a loose historical category associated with declarative logic, constraints and AI/expert systems. Generations do not determine whether an implementation uses an interpreter.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

MOE Computer Science Exit Exam · 2016 · Question 52

Question 7 | Q00071

What words are printed, in order, when execution enters this C++ fragment at label a and ends after its final statement? Ignore padding spaces.
a: cout<<"Graduating ";
goto b;
c: cout<<" Year ";
goto a;
b: cout<<"fourth ";

A. GraduatingYear Fourth

B. Graduating fourth

C. YearFourth Graduating

D. GraduatingFourth Year

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer B: B. Graduating fourth

Execution starts at a and prints Graduating. The unconditional goto b then transfers directly to the final output statement. That jump skips label c and its Year output, as well as the jump back to a. The statement at b prints lowercase fourth and the fragment ends. Ignoring padding spaces as specified, the two printed words are therefore Graduating followed by fourth.

Why each choice is right or wrong

A - Incorrect: Year is skipped, and Fourth changes the printed case.

B - Correct: The jump reaches b, giving Graduating followed by lowercase fourth.

C - Incorrect: Execution does not start at c, so Year cannot be the first word.

D - Incorrect: Year is never reached after b; the fragment ends there.

Primary note proof

Study-note section CP15

> A goto transfers to a label within its function; follow the label reached rather than reading lines in source order. In the sequence a: print Graduating; goto b; c: print Year; goto a; b: print fourth;, entry at a prints Graduating fourth and reaches the end. Year is not printed. Structured alternatives usually make such logic easier to reason about.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 14 | lines 15-20

> C++ goto Statement Syntax: goto label1; statements Label1: statements

Computer Programming module | PDF page 6 | lines 4-15

6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

MOE Computer Science Exit Exam · 2016 · Question 71

Question 8 | Q00074

Given the C++ declarations below, which statement correctly prints the car_model member through c1? Assume cout is available.
struct car { char car_model[25]; float price; };
car c{};
car* c1=&c;

A. Cout<<c1<<price;

B. Cin>>c1.car_model;

C. C1.price = 150000;

D. cout<<c1->car_model;

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer D: D. cout<<c1->car_model;

The variable c1 is a pointer to a live car object. Accessing a member through that pointer uses ->, followed by the exact member name car_model. The corrected D therefore inserts the character array into cout; zero initialization supplies a valid empty string here. The other expressions fail to select the requested member, use dot directly on a pointer, or refer to an undeclared differently capitalized name.

Why each choice is right or wrong

A - Incorrect: This does not select price from c1 and uses the undeclared name Cout.

B - Incorrect: A pointer needs -> rather than dot; Cin also has the wrong case.

C - Incorrect: C1 differs from c1, and dot is not pointer-member access.

D - Correct: cout<<c1->car_model; selects the declared member through the live pointer.

Primary note proof

Study-note section CP27

A struct groups named members, possibly of different types. struct House{double price; int rooms;}; defines a type; House h{150000,3}; creates an object. h.rooms uses a dot. House* p=&h; p->rooms is equivalent to (*p).rooms. The pointer must refer to a live object before member access. A type definition alone does not create an ordinary instance.

Study-note section CP-S01

> For an aggregate with no default member initializers, empty braces initialize its members recursively. Numeric scalar members become zero, and the elements of a character-array member become zero characters. A zero character at the beginning of such an array makes it an empty, terminated C string. This differs from leaving an ordinary automatic scalar or array uninitialized.

Study-note section CP24

> cout<<a prints a terminated char sequence, not its numeric address.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 38 | lines 10-16

> Pointer to structure Pointers to structures may be declared as follow: struct staructuretage *pinternname; eg Employe *emp; When using a pointer to a struct, member access can be achieved with the `.' operator, but can not look graceful in expression Example: (*emp).id Equivalently, the ->'` operator can be used; for example: emp->id

Computer Programming module | PDF page 35 | lines 13-17

> Accessing members of a structure variable A structure member can be accessed using dot operator'.' notation syntax: structvariable.member; eg. st.id=4;

Question sources

MOE Computer Science Exit Exam · 2016 · Question 74

Question 9 | Q00085

Software Debugging is known as _____

A. identifying the task to be computerized

B. creating program code

C. creating the algorithm

D. finding and correcting errors in the program code

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Finding and correcting errors in program code.

CP32 defines a bug as a defect and debugging as locating and correcting it. Defining the task, designing an algorithm and creating code are other development activities. They may expose defects, but none of them states the specific error-finding and correction purpose of debugging.

Why each choice is right or wrong

A - Incorrect: Identifying the task defines requirements; it does not locate and repair defects in code.

B - Incorrect: Creating code is implementation, whereas debugging investigates and corrects defects in that code.

C - Incorrect: Creating an algorithm is design; debugging concerns finding and correcting defects.

D - Correct: Finding and correcting program errors directly matches the definition of debugging in CP32.

Primary note proof

Study-note section CP32

> A bug is a defect; debugging locates and corrects it.

Study-note section CP01

> Programming includes writing, testing, debugging and maintaining instructions. A program can execute without solving its intended problem; compare its actual result with the required result.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 2-18

> 1. Introduction to programming Computer programming is the process of writing, testing, debugging/troubleshooting, and maintaining the source code of computer programs. Programs are the instructions that tells the computer what to do .so, writing computer programs means writing instructions, that will make the computer follow and run a program based on those instructions. We are going to write computer programs to solve problems. Problem Solving Problem: gap between the existing and the desired or proposed situation where problem solving will try to fill this gap. Problem solving: The process of transforming the description of a problem into the solution by using our knowledge of the problem-solving strategies, techniques and tools. Problem solving techniques • Analyze the problem: Carefully understand the problem • Outline the problem requirements • Does program require user interaction? • Does program manipulate data? • What is the output? • Design steps (algorithm) to solve the problem

Question sources

MOE Computer Science Exit Exam · 2016 · Question 85

MOE Computer Science Exit Exam · 2016 · Question 97

Question 10 | Q00099

How many times is the phrase "In the loop" printed when the following code is executed?
int i, j=8;
for(i=0; i<j; i++, j--)
3. {
if(i%2!=0) continue;
cout << "In the loop " << endl;
6. }

A. 20

B. 10

C. 3

D. 2

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: 2

Ignoring source line numbers, the loop body is reached at (i,j)=(0,8),(1,7),(2,6),(3,5). Even i values 0 and 2 print; odd values 1 and 3 continue to the update. The next pair (4,4) fails i<j. Therefore exactly two lines print, not 20, 10 or 3. Module support: PDF p. 12, PDF p. 13, PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "20" does not match the rule or result derived here. Ignoring source line numbers, the loop body is reached at (i,j)=(0,8),(1,7),(2,6),(3,5). Even i values 0 and 2 print; odd values 1 and 3 continue to the update. The next pair (4,4) fails i<j. Therefore exactly two lines print, not 20, 10 or 3.

B - Incorrect: "10" does not match the rule or result derived here. Ignoring source line numbers, the loop body is reached at (i,j)=(0,8),(1,7),(2,6),(3,5). Even i values 0 and 2 print; odd values 1 and 3 continue to the update. The next pair (4,4) fails i<j. Therefore exactly two lines print, not 20, 10 or 3.

C - Incorrect: "3" does not match the rule or result derived here. Ignoring source line numbers, the loop body is reached at (i,j)=(0,8),(1,7),(2,6),(3,5). Even i values 0 and 2 print; odd values 1 and 3 continue to the update. The next pair (4,4) fails i<j. Therefore exactly two lines print, not 20, 10 or 3.

D - Correct: "2" matches the derived result. Ignoring source line numbers, the loop body is reached at (i,j)=(0,8),(1,7),(2,6),(3,5). Even i values 0 and 2 print; odd values 1 and 3 continue to the update. The next pair (4,4) fails i<j. Therefore exactly two lines print, not 20, 10 or 3.

Primary note proof

Study-note section CP16

Make a table with condition inputs, body output and updated values. For i=0,j=8; i<j; i++,j--, body entries are (0,8),(1,7),(2,6),(3,5). If odd i triggers continue, printing occurs twice. With outer i=1; i<5; i+=5, the outer loop runs once; an inner j=10; j>i; --j prints nine times.

Study-note section CP15

continue skips the rest of the current loop iteration: a for loop then runs its update; a while loop goes directly to its condition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 19-28

The for loop. Its format is: for (initialization; condition; increase) statement; 1, initialization is executed. Generally, it is a initial value setting for a counter variable. This is executed only once 2, condition is checked, if it is true the loop continues, otherwise the loop finishes and statement is skipped. 3, statement is executed. As usual, it can be either a single instruction or a block of instructions enclosed within curly brackets { }. 4, finally, whatever is specified in the increase field is executed and the loop gets back to step 2.

Computer Programming module | PDF page 13 | lines 30-35

Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Addis Ababa University mock exam · Question 1

Question 11 | Q00100

What does this C++ fragment print? Assume the standard streams are available; line breaks shown in the options are part of the output.

```
int main(){
int a=5,x=1,y=0,z=4;
a=(x && y) || ++z;
std::cout<<z<<std::endl<<a;
return 0;
}
```

A. 5
1

B. 5
2

C. 4
1

D. 4
2

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer A: A. 5, then 1 on the next line.

The conjunction x&&y is false because y is zero. The OR therefore evaluates ++z, which changes z from 4 to 5 and supplies a true value. Assigning that logical result stores 1 in a. The output inserts z, then a line ending, then a, so it consists of 5 followed by 1 on the next line.

Why each choice is right or wrong

A - Correct: The false left operand triggers ++z, and the logical result is 1.

B - Incorrect: A logical OR produces 0 or 1, so a cannot print as 2.

C - Incorrect: The OR evaluates ++z, so z does not remain 4.

D - Incorrect: This both omits the increment and gives the logical result as 2.

Primary note proof

Study-note section CP10

> && evaluates its right operand only if the left is true; || only if the left is false. Thus false && dangerousCall() skips the call. Bitwise & operates on bits and does not short circuit. With x=1,y=0,z=4, (x&y)||++z evaluates ++z, leaving z=5 and the Boolean result 1. The precedence ladder is grouping/postfix, unary, multiplicative, additive, shifts, relations, equality, bitwise &, ^, |, &&, ||, conditional, assignment, comma.

Study-note section CP28

std::endl inserts a newline and flushes.

Study-note section CP34

With valid C++ syntax for the stated assignments, the trace yields z=5 then a=1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 6 | lines 4-15

6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Addis Ababa University mock exam · Question 2

Question 12 | Q00101

is a function that invokes itself?

A. Recursive function

B. Inline function

C. Built in function

D. User defined function

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: Recursive function

A recursive function calls itself, exactly the definition on p.18. Inline concerns implementation and built-in/user-defined concern origin; none of those categories alone entails self-invocation. Module support: PDF p. 18.

Why each choice is right or wrong

A - Correct: "Recursive function" matches the derived result. A recursive function calls itself, exactly the definition on p.18. Inline concerns implementation and built-in/user-defined concern origin; none of those categories alone entails self-invocation.

B - Incorrect: "Inline function" does not match the rule or result derived here. A recursive function calls itself, exactly the definition on p.18. Inline concerns implementation and built-in/user-defined concern origin; none of those categories alone entails self-invocation.

C - Incorrect: "Built in function" does not match the rule or result derived here. A recursive function calls itself, exactly the definition on p.18. Inline concerns implementation and built-in/user-defined concern origin; none of those categories alone entails self-invocation.

D - Incorrect: "User defined function" does not match the rule or result derived here. A recursive function calls itself, exactly the definition on p.18. Inline concerns implementation and built-in/user-defined concern origin; none of those categories alone entails self-invocation.

Primary note proof

Study-note section CP21

> A recursive function calls itself directly or indirectly.

Study-note section CP19

> inline does not compel the compiler to copy a function body at a call. Small frequently used functions may be good optimization candidates, but actual expansion is the optimizer’s decision. Inline also has language linkage/definition rules, so it is not simply a speed command.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 18 | lines 23-28

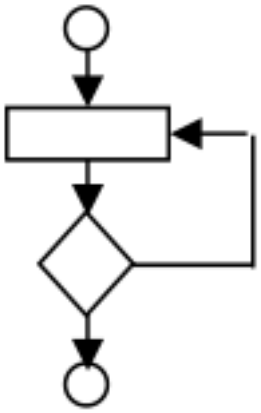
> Recursive Functions - Functions that call themselves - A function that is defined in terms of itself is called recursive - There must be a base case e.g. 0! = 1 - Slowly converges towards base case - For example, Recursive functions N! = N * (N-1)!

Question sources

Addis Ababa University mock exam · Question 3

Question 13 | Q00102

Which control flow statement is represented by the flowchart shown in the figure below?



A. while loop

B. do...while loop

C. if...else statement

D. for loop

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: do...while loop

The retained flowchart enters the rectangular body before reaching the decision diamond. Its backward arrow returns from the decision to the body. This is a post-test repetition with at least one execution, matching do-while (p.12). while and ordinary for test before the body; if-else does not loop back. Module support: PDF p. 12, PDF p. 11.

Why each choice is right or wrong

A - Incorrect: "while loop" does not match the rule or result derived here. The retained flowchart enters the rectangular body before reaching the decision diamond. Its backward arrow returns from the decision to the body. This is a post-test repetition with at least one execution, matching do-while (p.12). while and ordinary for test before the body; if-else does not loop back.

B - Correct: "do...while loop" matches the derived result. The retained flowchart enters the rectangular body before reaching the decision diamond. Its backward arrow returns from the decision to the body. This is a post-test repetition with at least one execution, matching do-while (p.12). while and ordinary for test before the body; if-else does not loop back.

C - Incorrect: "if...else statement" does not match the rule or result derived here. The retained flowchart enters the rectangular body before reaching the decision diamond. Its backward arrow returns from the decision to the body. This is a post-test repetition with at least one execution, matching do-while (p.12). while and ordinary for test before the body; if-else does not loop back.

D - Incorrect: "for loop" does not match the rule or result derived here. The retained flowchart enters the rectangular body before reaching the decision diamond. Its backward arrow returns from the decision to the body. This is a post-test repetition with at least one execution, matching do-while (p.12). while and ordinary for test before the body; if-else does not loop back.

Primary note proof

Study-note section CP14

> while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Study-note section CP34

> When entry reaches the body rectangle before the decision diamond and a decision branch returns to the body, the body executes before the condition is tested. This is a post-test do-while loop.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 2-7

> The do-while loop The condition in the do-while is evaluated after the execution of statement instead of before, granting at least one execution of statement even if condition is never fulfilled. do {statement; }while (condition); First statement is executed and then condition is evaluated. If the outcome of the latter is nonzero then the whole process is repeated. Otherwise, the loop is terminated.

Computer Programming module | PDF page 11 | lines 5-13

> Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Question sources

Addis Ababa University mock exam · Question 4

Question 14 | Q00103

What is the output of the following code snippet?
double ph=15.3;
if(ph<@ || ph>15)
3. {
cout<< "Invalid PH value. ";
5. }
if(ph<=6.5)
cout<< "Acidic";
else if(ph>6.5 && ph<7.5)
cout<< "Neutral";
else
cout<< "Alkaline";

A. Invalid PH Value

B. Neutral

C. Alkaline

D. Invalid PH Value. Alkaline

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: Invalid PH Value. Alkaline

The negative-value test is ph<0. Since 15.3>15, the first independent if prints Invalid PH value. The following separate if/else-if chain still executes: ph<=6.5 is false and the middle range test is false, so it prints Alkaline. Thus both messages appear. A and C omit one message, and B takes a false middle branch. Module support: PDF p. 9, PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "Invalid PH Value" does not match the rule or result derived here. The negative-value test is ph<0. Since 15.3>15, the first independent if prints Invalid PH value. The following separate if/else-if chain still executes: ph<=6.5 is false and the middle range test is false, so it prints Alkaline. Thus both messages appear. A and C omit one message, and B takes a false middle branch.

B - Incorrect: "Neutral" does not match the rule or result derived here. The negative-value test is ph<0. Since 15.3>15, the first independent if prints Invalid PH value. The following separate if/else-if chain still executes: ph<=6.5 is false and the middle range test is false, so it prints Alkaline. Thus both messages appear. A and C omit one message, and B takes a false middle branch.

C - Incorrect: "Alkaline" does not match the rule or result derived here. The negative-value test is ph<0. Since 15.3>15, the first independent if prints Invalid PH value. The following separate if/else-if chain still executes: ph<=6.5 is false and the middle range test is false, so it prints

Alkaline. Thus both messages appear. A and C omit one message, and B takes a false middle branch.

D - Correct: "Invalid PH Value. Alkaline" matches the derived result. The negative-value test is ph<0. Since 15.3>15, the first independent if prints Invalid PH value. The following separate if/else-if chain still executes: ph<=6.5 is false and the middle range test is false, so it prints Alkaline. Thus both messages appear. A and C omit one message, and B takes a false middle branch.

Primary note proof

Study-note section CP12

A standalone if may execute independently of the following if. An else binds to its nearest unmatched if; an else-if ladder executes the first true branch only. In nested if(c1){if(c2) S1; else S2;}, S2 requires c1 true and c2 false. Braces determine grouping.

Study-note section CP12

With ph=15.3, an invalid-range if can print its warning, and a separate classification ladder can also print Alkaline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 9 | lines 9-21

Conditional Statements A. The if Statement Syntax: if (expression) statement; First expression is evaluated. If the outcome is nonzero (true) then statement is executed. Otherwise, nothing happens. if-else statement Syntax: if (expression) statement1; else statement2; First expression is evaluated. If the outcome is nonzero (true) then statement1 is executed. Otherwise, statement2 is executed.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Addis Ababa University mock exam · Question 5

Question 15 | Q00104

Which three numeric values are displayed, in order, by this C++ program? Ignore the prompt text and separator spaces; assume cout and endl are available.

```
int func(int a,int b){
a*=5;
b+=5;
return a+b;
}
int main(){
int x=9,y=3;
cout<<"Result = "<<func(x,y)<<", "<<x<<", "<<y<<endl;
return 0;
}
```

A. 12,9,3

B. 50,9,3

C. 53,9,3

D. 53,45,8

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: C. 53,9,3

The value parameters are separate objects initialized with 9 and 3. Inside func, a becomes 45 and b becomes 8, so the returned sum is 53. Neither update changes the caller's x or y, which remain 9 and 3. The requested numeric sequence is therefore 53,9,3. Standard int main() restores the valid enclosing program already required by CP34 without changing the parameter-passing problem.

Why each choice is right or wrong

A - Incorrect: 12 adds the original parameters instead of their updated local values.

B - Incorrect: The updated local values sum to 53, not 50.

C - Correct: The function returns 53 while the caller retains 9 and 3.

D - Incorrect: Value parameters do not change the caller's x and y to 45 and 8.

Primary note proof

Study-note section CP18

> A C++ value parameter is a separate object initialized from the argument. In func(int a,int b){a*=5;b+=5;return a+b;}, calling func(9,3) returns 53 and leaves the caller values 9 and 3 unchanged.

Study-note section CP34

Use <0 for a less-than-zero comparison and the standard int main() declaration for a C++ entry point.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 16 | lines 13-20

Passing by Value In call by value, the actual value that is passed as argument is not changed after performing some operation on it. it creates a copy of that variable into the stack section in memory. When the value is changed, it changes the value of that copy, the actual value remains the same Example: Passing by Reference: It allows a function to modify a variable without having to create a copy of it. We have to declare reference variables. The memory location of the passed variable and parameter is the same and therefore, any change to the parameter reflects in the variable as well.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, | |) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 | | 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Addis Ababa University mock exam · Question 6

Question 16 | Q00105

What do arrays do?

A. Hold address of a single value

B. Hold one value

C. Hold values of the same type under a single name

D. Hold addresses of values under a single name

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Hold values of the same type under a single name

An array groups a fixed number of elements of the same type (p.19), with each element selected by an index. It is not a single-value storage definition or intrinsically a collection of addresses; pointer-valued arrays are only a particular element type. Module support: PDF p. 19.

Why each choice is right or wrong

A - Incorrect: "Hold address of a single value" does not match the rule or result derived here. An array groups a fixed number of elements of the same type (p.19), with each element selected by an index. It is not a single-value storage definition or intrinsically a collection of addresses; pointer-valued arrays are only a particular element type.

B - Incorrect: "Hold one value" does not match the rule or result derived here. An array groups a fixed number of elements of the same type (p.19), with each element selected by an index. It is not a single-value storage definition or intrinsically a collection of addresses; pointer-valued arrays are only a particular element type.

C - Correct: "Hold values of the same type under a single name" matches the derived result. An array groups a fixed number of elements of the same type (p.19), with each element selected by an index. It is not a single-value storage definition or intrinsically a collection of addresses; pointer-valued arrays are only a particular element type.

D - Incorrect: "Hold addresses of values under a single name" does not match the rule or result derived here. An array groups a fixed number of elements of the same type (p.19), with each element selected by an index. It is not a single-value storage definition or intrinsically a collection of addresses; pointer-valued arrays are only a particular element type.

Primary note proof

Study-note section CP22

> A built-in C++ array has a fixed number of same-type elements in contiguous storage: int a[4]={10,20,30,40};. Valid indices are 0 through 3. a[4] is outside the array; accessing it has undefined behavior. Java checks bounds and throws ArrayIndexOutOfBoundsException. Do not predict a C++ out-of-range value from nearby memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 19 | lines 4-29

> Arrays - an array is a collection of fixed-size number of elements of the same type stored sequentially in memory. - An individual element of an array is identified by its own unique index (or subscript). - The number of elements must be an integer. - The size of the array is referred to as its dimension - Array in C++ is Zero indexed. Declaring of one-dimensional Array Syntax: elementstype arrayname [size]; → You can use constants such as macro definitions or const ints as the sizes of arrays. Example: int num[5]; string fruit[3]; #define numstudents 100 or const int numstudents=100; double Score[numstudents] ; double Score[numstudents] ; Initialization of an Array - Arrays can be initialized at declaration time in two different ways: Syntax: elementstype arrayname[size] = {list each value separated by comma}; E.g int num[5]={1,2,3,4,5}; elementstype arrayname[ ] = { list with any number of values }; E.g int num[]={1,2,3,4,5}; string

fruit[]={“Apple”,“orange”, “Mango”}; Accessing Array Elements - Each element could be accessed using an index into the array - When accessing an array, the index given must be a positive integer from 0 to n-1, where n is the dimension of the array.

Question sources

Addis Ababa University mock exam · Question 7

Question 17 | Q00174

In recursion, the condition for which the function will stop calling itself is .

A. Best case

B. Worst case

C. Base case

D. There is no such condition

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Base case

The recursion section explicitly requires a base case and convergence toward it. That case stops further self-calls. Best and worst case describe other analyses, while denying a stopping condition conflicts with the module requirement. Module support: PDF p. 18.

Why each choice is right or wrong

A - Incorrect: "Best case" does not match the rule or result derived here. The recursion section explicitly requires a base case and convergence toward it. That case stops further self-calls. Best and worst case describe other analyses, while denying a stopping condition conflicts with the module requirement.

B - Incorrect: "Worst case" does not match the rule or result derived here. The recursion section explicitly requires a base case and convergence toward it. That case stops further self-calls. Best and worst case describe other analyses, while denying a stopping condition conflicts with the module requirement.

C - Correct: "Base case" matches the derived result. The recursion section explicitly requires a base case and convergence toward it. That case stops further self-calls. Best and worst case describe other analyses, while denying a stopping condition conflicts with the module requirement.

D - Incorrect: "There is no such condition" does not match the rule or result derived here. The recursion section explicitly requires a base case and convergence toward it. That case stops further self-calls. Best and worst case describe other analyses, while denying a stopping condition conflicts with the module requirement.

Primary note proof

Study-note section CP21

> A recursive function calls itself directly or indirectly. It needs a base case and progress toward it for all supported inputs. factorial(0)=1 and factorial(n)=n*factorial(n-1) for positive n. factorial(3) expands to 3*2*1*1=6. Negative inputs require rejection or a separate definition; they do not reach zero under n-1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 18 | lines 23-28

> Recursive Functions - Functions that call themselves - A function that is defined in terms of itself is called recursive - There must be a base case e.g. 0! = 1 - Slowly converges towards base case - For example, Recursive functions N! = N * (N-1)!

Question sources

Addis Ababa University mock exam · Question 76

Question 18 | Q00209

One of the following statements is true

A. The typedef command is used to create a new type.

B. If the function is declared inline the compiler must copy the code in the calling function

C. All of the choices are correct

D. It is recommended to use inline functions to simple and frequently used functions

E. None of the choices are correct

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Small, frequently used functions are reasonable inline optimization candidates.

CP19 recommends small, frequently used functions as possible optimization candidates, while stressing that inline never forces expansion at the call site. CP06 says typedef creates an alias rather than a distinct new type. Thus D is defensible as a recommendation; the mandatory-expansion and distinct-type claims cannot make all choices correct.

Why each choice is right or wrong

A - Incorrect: A typedef supplies an alternative name for an existing type, not a distinct new type.

B - Incorrect: Declaring inline does not require the compiler to expand the body at each call.

C - Incorrect: The distinct-type and mandatory-expansion claims are false, so all choices cannot be correct.

D - Correct: Simple, frequently called functions can be reasonable candidates, although expansion remains an optimizer decision.

E - Incorrect: The recommendation in D is supported, so it is false that none is correct.

Primary note proof

Study-note section CP19

> inline does not compel the compiler to copy a function body at a call. Small frequently used functions may be good optimization candidates, but actual expansion is the optimizer’s decision. Inline also has language linkage/definition rules, so it is not simply a speed command.

Study-note section CP06

> typedef creates an alias, not a distinct new type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 29-46

> • User-defined - Function that been created by the user. - These functions need to be declared and defined by the user. Declaration of function Function defines using the following syntax: Function_type function_name (Parameter List) { ...function body } Example: void Addition(int num1,int num2) #include<iostream> using namespace std; void Addition(int num1,int num2) { int sum=num1+num2; cout<<"The sum is:"<<sum; } int main() { int x=10,y=20; Addition(x,y);

Question sources

Mock exam 4 · Question 11

Question 19 | Q00210

Given the following lines of code, how many times will "hello" be displayed on the screen?

```
int i = 1;
while(i < 5){
int j = 10;
while (j > i){
j--;
cout << "hello" << endl;
}
i = i + 5;
}
```

A. 10

B. 9

C. 5

D. 25

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: 9

The outer loop starts with i=1. The inner loop decrements j from 10 through 1 and prints after each decrement: nine prints. Then i becomes 6, failing i<5, so no second outer iteration occurs. Ten counts the initial j without a successful iteration; five and 25 ignore the actual bounds. Module support: PDF p. 11, PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "10" does not match the rule or result derived here. The outer loop starts with i=1. The inner loop decrements j from 10 through 1 and prints after each decrement: nine prints. Then i becomes 6, failing i<5, so no second outer iteration occurs. Ten counts the initial j without a successful iteration; five and 25 ignore the actual bounds.

B - Correct: "9" matches the derived result. The outer loop starts with i=1. The inner loop decrements j from 10 through 1 and prints after each decrement: nine prints. Then i becomes 6, failing i<5, so no second outer iteration occurs. Ten counts the initial j without a successful iteration; five and 25 ignore the actual bounds.

C - Incorrect: "5" does not match the rule or result derived here. The outer loop starts with i=1. The inner loop decrements j from 10 through 1 and prints after each decrement: nine prints. Then i becomes 6, failing i<5, so no second outer iteration occurs. Ten counts the initial j without a successful iteration; five and 25 ignore the actual bounds.

D - Incorrect: "25" does not match the rule or result derived here. The outer loop starts with i=1. The inner loop decrements j from 10 through 1 and prints after each decrement: nine prints. Then i becomes 6, failing i<5, so no second outer iteration occurs. Ten counts the initial j without a successful iteration; five and 25 ignore the actual bounds.

Primary note proof

Study-note section CP14

> while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Study-note section CP16

> With outer i=1; i<5; i+=5, the outer loop runs once; an inner j=10; j>i; --j prints nine times.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 11 | lines 5-13

> Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While

statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Mock exam 4 · Question 12

Question 20 | Q00211

What is the output of the below given program?

```
int figure_me_out() {
static int x = 0;
if (x % 2 == 0)
x = x + 1;
else
x = x + 2;
return x;
}
int main() {
int x = 0;
for(int i = 0; i < 5; i++)
x = x + figure_me_out();
cout << x;
return 0;
}
```

A. 1

B. 19

C. 2

D. 20

E. 25

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer E: 25.

The function's static x is initialized once, then persists across calls. Its five returns are 1, 3, 5, 7 and 9: after the first update it stays odd and increases by two. Main has a separate x and accumulates all five returns, giving 25.

Why each choice is right or wrong

A - Incorrect: One is only the first function return, not the sum accumulated by main.

B - Incorrect: The five return values sum to 25; a total of 19 omits or miscomputes updates.

C - Incorrect: The function does not reset or simply count calls; its persistent values are odd.

D - Incorrect: The five odd returns total 25, not the sum of five even values.

E - Correct: Adding the five persistent-state returns gives 1+3+5+7+9, which equals 25.

Primary note proof

Study-note section CP20

> An automatic local is created anew on entry. A static local is initialized once and retains its value across calls even though its name remains local. If static x=0 changes by +1 when even and +2 otherwise, five calls return 1,3,5,7,9, whose sum is 25. Namespace/static storage integers are zero initialized; an ordinary uninitialized local int has no usable assumed zero value.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 17 | lines 14-29

> Scope of variables Scope of variables defined as the extent of the program code within which the variable can be accessed or declared or worked with mainly two types of variable scopes: local variable: - Variables defined within a function or block are said to be local to those functions - Declared inside a block{} Global Variable - variable that can be accessed from any part of the program. - They are available throughout the life time of a program. - They are declared at the top of the program outside all of the functions or blocks. NB: - Usually when two variables with same name are defined then the compiler produces a compile time error. But if the variables are defined in different scopes, then the compiler allows it. - Whenever there is a local variable defined with same name as that of a global variable then the compiler will give precedence to the local variable

Computer Programming module | PDF page 11 | lines 5-13

> Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then

statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Question sources

Mock exam 4 · Question 13

Question 21 | Q00212

What is the output of the following code fragment?

```
int f1(int n, int m){
if(n < m)
return 0;
else if(n == m)
return m + f1(n-1, m);
else
return n + f1(n-2, m-1);
}
int main(){
cout << f1(1, 4);
return 0;
}
```

A. 0

B. None of the choices are correct

C. 8

D. 4

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer A: 0

The initial call is f1(1,4). Its first condition 1<4 is true, so it returns 0 immediately. Neither recursive branch executes. Eight and four would require branches that are not entered; the none option is false because 0 is listed. Module support: PDF p. 18, PDF p. 9.

Why each choice is right or wrong

A - Correct: "0" matches the derived result. The initial call is f1(1,4). Its first condition 1<4 is true, so it returns 0 immediately. Neither recursive branch executes. Eight and four would require branches that are not entered; the none option is false because 0 is listed.

B - Incorrect: "None of the choices are correct" does not match the rule or result derived here. The initial call is f1(1,4). Its first condition 1<4 is true, so it returns 0 immediately. Neither recursive branch executes. Eight and four would require branches that are not entered; the none option is false because 0 is listed.

C - Incorrect: "8" does not match the rule or result derived here. The initial call is f1(1,4). Its first condition 1<4 is true, so it returns 0 immediately. Neither recursive branch executes. Eight and four would require branches that are not entered; the none option is false because 0 is listed.

D - Incorrect: "4" does not match the rule or result derived here. The initial call is f1(1,4). Its first condition 1<4 is true, so it returns 0 immediately. Neither recursive branch executes. Eight and four would require branches that are not entered; the none option is false because 0 is listed.

Primary note proof

Study-note section CP21

> Evaluate conditions before expanding a recursive call. For if(n<m) return 0;, f(1,4) returns immediately; unreachable recursive branches add nothing. Record each call's local parameters and pending operation, then unwind returned values. Recursion commonly uses smaller instances, but merely having a base-case branch does not prove it will be reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 18 | lines 23-28

> Recursive Functions - Functions that call themselves - A function that is defined in terms of itself is called recursive - There must be a base case e.g. 0! = 1 - Slowly converges towards base case - For example, Recursive functions N! = N * (N-1)!

Computer Programming module | PDF page 9 | lines 9-21

> Conditional Statements A. The if Statement Syntax: if (expression) statement; First expression is evaluated. If the outcome is nonzero (true) then statement is executed. Otherwise, nothing happens. if-else statement Syntax: if (expression) statement1; else statement2; First expression is evaluated. If the outcome is nonzero (true) then statement1 is executed. Otherwise, statement2 is executed.

Question sources

Mock exam 4 · Question 14

Question 22 | Q00270

Assume that p, x, y, and z are int variables with x = 1, y = 10, z= -3. Which of the following is true after the following statement is executed? p= (x++ * y--) + ++z;

A. p=22

B. p=17

C. p=8

D. p=16

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: p=8

Postfix x++ contributes old x=1 and y-- contributes old y=10; prefix ++z changes -3 to -2 before use. Thus p=1*10+(-2)=8. The new x=2 and y=9 are not the values multiplied. The other values mix old and updated operands. Module support: PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "p=22" does not match the rule or result derived here. Postfix x++ contributes old x=1 and y-- contributes old y=10; prefix ++z changes -3 to -2 before use. Thus p=1*10+(-2)=8. The new x=2 and y=9 are not the values multiplied. The other values mix old and updated operands.

B - Incorrect: "p=17" does not match the rule or result derived here. Postfix x++ contributes old x=1 and y-- contributes old y=10; prefix ++z changes -3 to -2 before use. Thus p=1*10+(-2)=8. The new x=2 and y=9 are not the values multiplied. The other values mix old and updated operands.

C - Correct: "p=8" matches the derived result. Postfix x++ contributes old x=1 and y-- contributes old y=10; prefix ++z changes -3 to -2 before use. Thus p=1*10+(-2)=8. The new x=2 and y=9 are not the values multiplied. The other values mix old and updated operands.

D - Incorrect: "p=16" does not match the rule or result derived here. Postfix x++ contributes old x=1 and y-- contributes old y=10; prefix ++z changes -3 to -2 before use. Thus p=1*10+(-2)=8. The new x=2 and y=9 are not the values multiplied. The other values mix old and updated operands.

Primary note proof

Study-note section CP09

> Prefix ++x changes x and yields the new value; postfix x++ yields the previous value and also increments x. Separate the value used in the expression from the stored value after it. Starting a=8,b=5, a++ + --b uses 8+4, gives 12, and leaves a=9,b=4. Starting x=1,y=10,z=-3, (x++*y--)+ ++z gives 8 and leaves x=2,y=9,z=-2.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science model exam · Question 1

University of Gondar model exam · Question 1

Question 23 | Q00271

What will happen in the following C++ code snippet? int a = 100, b = 200; int *p = &a, *q = &b; p= q;

A. p now points to b

B. a is assigned to b

C. b is assigned to a

D. q now points to a

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: p now points to b

p initially stores &a and q stores &b. Assignment p=q copies the address stored in q into p, so both point to b. Neither pointee value is assigned because no dereference appears; q is not assigned at all. Module support: PDF p. 26, PDF p. 27.

Why each choice is right or wrong

A - Correct: "p now points to b" matches the derived result. p initially stores &a and q stores &b. Assignment p=q copies the address stored in q into p, so both point to b. Neither pointee value is assigned because no dereference appears; q is not assigned at all.

B - Incorrect: "a is assigned to b" does not match the rule or result derived here. p initially stores &a and q stores &b. Assignment p=q copies the address stored in q into p, so both point to b. Neither pointee value is assigned because no dereference appears; q is not assigned at all.

C - Incorrect: "b is assigned to a" does not match the rule or result derived here. p initially stores &a and q stores &b. Assignment p=q copies the address stored in q into p, so both point to b. Neither pointee value is assigned because no dereference appears; q is not assigned at all.

D - Incorrect: "q now points to a" does not match the rule or result derived here. p initially stores &a and q stores &b. Assignment p=q copies the address stored in q into p, so both point to b. Neither pointee value is assigned because no dereference appears; q is not assigned at all.

Primary note proof

Study-note section CP25

> int x=10; int* p=&x; stores x's address in p. *p reads or writes x; &p is the address of the pointer variable itself. With p=&a,q=&b, p=q redirects p to b and leaves a and b unchanged. By contrast *p=*q copies the pointed-to value. Never dereference an uninitialized, null or dangling pointer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 26 | lines 2-22

> Pointer - A pointer is a variable that holds a memory address of another variable. - - Its content is a memory address - A pointer is simply the address of a memory location and provides an indirect way of accessing data in memory. - It is defined to „point to“ data of a specific type. - Because a pointer variable holds the address of another piece of data, it "points to" the data Reference operator (&) and Dereference operator (*) &: read as address of - Use address operator & to get address of a variable *: read as value pointed by Declaring variables of pointer types Syntax: type *variable_name; For example: int *ptr1; //pointertoanint double*number; char*ptr2; //pointertoachar The value of a pointer variable is the address to which it points. For example, given the definitions int num; wecanwrite: ptr1=#

Computer Programming module | PDF page 27 | lines 21-35

> int num1 = 5, num2 = 15; int *p1, *p2; p1 = &num1; p2 = &num2; cout<<*p1<<endl; *p1 = 10; cout<<"num1:"<<num1<<endl; *p2 = *p1; cout<<*p2<<endl; p1 = p2; cout<<*p1<<endl; *p1 = 20; cout<<"num1:"<< num1 << endl; cout<<"num2:"<<num2 << endl; cout<<*p2<<" "<<*p1;

Question sources

Computer Science model exam · Question 2

Computer Science model exam · Question 7

University of Gondar model exam · Question 2

University of Gondar model exam · Question 7

Question 24 | Q00272

Which of the following best describes the statement "C++ is a case sensitive programming language"?

A. All variables in C++ are identifiers

B. Every C++ statement should include the header file iostream.h

C. ExitExam and exitExam are distinct in C++

D. Every statement in C++ ends with semicolon

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: ExitExam and exitExam are distinct identifiers.

Case sensitivity means changing uppercase to lowercase can change the identifier. ExitExam and exitExam therefore name different entities in C++. Claims about headers, variable terminology or semicolon placement do not explain case sensitivity; the universal header and semicolon assertions also overstate C++ syntax requirements.

Why each choice is right or wrong

A - Incorrect: Variable names are identifiers, but this does not describe the effect of changing letter case.

B - Incorrect: Header inclusion is unrelated to case sensitivity, and headers are not included in every statement.

C - Correct: The differing initial letter case makes ExitExam and exitExam distinct C++ identifiers.

D - Incorrect: Semicolon rules concern grammar, not identifier case, and do not apply uniformly to every construct.

Primary note proof

Study-note section CP05

> In the basic ASCII identifier rules, start with a letter or underscore; continue with letters, digits or underscores. Do not use a keyword. C++ is case sensitive: exitExam and ExitExam differ; delete and float are keywords, but FLOAT is a legal identifier. variable_1234 is valid; 4thInput, T-3 and names containing spaces are not single identifiers. Avoid reserved underscore patterns in application code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 2-10

> 1. Keywords (reserved words) - Unique meaning within a C++ program and must not be used for any other purposes. E.g., if, namespace for, switch, continue, float, new, default, break Etc. 2. Identifiers - Name associated with a function or data object and used to refer to that function or data object. An identifier must: o Start with a letter or underscore o Consist only of letters, the digits 0-9, or the underscore symbol _ o Not be a reserved word

Question sources

Computer Science model exam · Question 3

University of Gondar model exam · Question 3

Question 25 | Q00273

Assume a and b are distinct ordinary int variables and b has an initialized value. Immediately after a=b;, which statements about the assignment and subsequent changes to b are true? Select all that apply.

A. The value of a and b are equal

B. The value of b is assigned to variable a and the later change on variable b will affect the value of variable a

C. The value of variable a is assigned to variable b and the value of variable b is assigned to variable a

D. The value of b is assigned to variable a and the later change on variable b will not affect the value of variable a

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A, D: A. The value of a and b are equal; D. The value of b is assigned to variable a and the later change on variable b will not affect the value of variable a

Scalar assignment copies the value stored in b into the separate object a. Immediately afterward, the two values are equal. Their storage remains independent, however: changing b later does not automatically update a. Thus both the immediate-equality description and the copy-with-independence description are true. Assignment works in one direction and is neither a swap nor a reference binding between these two int variables.

Why each choice is right or wrong

A - Correct: The copied int values are equal immediately after assignment.

B - Incorrect: Separate scalar storage means later changes to b do not affect a.

C - Incorrect: Assignment copies one way; it does not swap values.

D - Correct: This correctly states both the copying direction and later independence.

Primary note proof

Study-note section CP06

> Write type name; or type name = value; for ordinary variables: int age = 20;. Declaration introduces a name and type; initialization gives its initial value; later assignment replaces a value. a=b copies a scalar value, so later changes to b do not change a. A pointer or Java object reference instead copies an address/reference.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 5 | lines 24-33

5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Question sources

Computer Science model exam · Question 4

University of Gondar model exam · Question 4

Question 26 | Q00274

Which one of the following is true statement about an algorithm?

A. Ambiguous steps are allowed in an algorithm.

B. Complex problems cannot have an algorithm.

C. An algorithm may not have a beginning and an end.

D. An algorithm should take finite amount of time to compute.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: An algorithm should take finite amount of time to compute.

The module defines an algorithm as unambiguous executable steps forming a terminating process and explicitly requires finite execution for each input. A permits ambiguity, B imposes an unsupported complexity restriction, and C allows missing termination, contrary to that definition. Module support: PDF p. 2.

Why each choice is right or wrong

A - Incorrect: "Ambiguous steps are allowed in an algorithm." does not match the rule or result derived here. The module defines an algorithm as unambiguous executable steps forming a terminating process and explicitly requires finite execution for each input. A permits ambiguity, B imposes an unsupported complexity restriction, and C allows missing termination, contrary to that definition.

B - Incorrect: "Complex problems cannot have an algorithm." does not match the rule or result derived here. The module defines an algorithm as unambiguous executable steps forming a terminating process and explicitly requires finite execution for each input. A permits ambiguity, B imposes an unsupported complexity restriction, and C allows missing termination, contrary to that definition.

C - Incorrect: "An algorithm may not have a beginning and an end." does not match the rule or result derived here. The module defines an algorithm as unambiguous executable steps forming a terminating process and explicitly requires finite execution for each input. A permits ambiguity, B imposes an unsupported complexity restriction, and C allows missing termination, contrary to that definition.

D - Correct: "An algorithm should take finite amount of time to compute." matches the derived result. The module defines an algorithm as unambiguous executable steps forming a terminating process and explicitly requires finite execution for each input. A permits ambiguity, B imposes an unsupported complexity restriction, and C allows missing termination, contrary to that definition.

Primary note proof

Study-note section CP02

> An algorithm gives precise, executable steps and terminates for every legal input. Definiteness means each step has an unambiguous meaning; effectiveness means a step can actually be performed. Inputs may be absent, but a useful algorithm specifies a result. Correctness means the result meets the specification. Efficiency is desirable and measurable, but an inefficient terminating algorithm is still an algorithm. Efficiency is also a desirable property of an algorithm.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 19-26

> Algorithm is an ordered set of unambiguous, executable steps, defining a terminating process. Important Properties of Algorithms - Correct: always returns the desired output for all legal instances of the problem. - Unambiguous: Each step must be clearly defined, having only one interpretation. - Precise: simple but efficient - Efficient: can be measured in terms of Time and Space - Finiteness: the execution of the algorithm must terminate in a finite amount of time for each possible set of input data/must have an end.

Question sources

Computer Science model exam · Question 5

University of Gondar model exam · Question 5

Question 27 | Q00275

Which of the following will NOT increase an integer variable "i" by 1?

A. i+=1;

B. i=i+1; E. ++i;

C. i=i+i;

D. i++;

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: i=i+i;

i=i+i doubles i rather than adding a fixed one. i+=1, i=i+1, i++ and the embedded ++i all increment by one. At the special value i=1 doubling also adds one; the intended comparison is between operations for general i. Module support: PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "i+=1;" does not match the rule or result derived here. i=i+i doubles i rather than adding a fixed one. i+=1, i=i+1, i++ and the embedded ++i all increment by one. At the special value i=1 doubling also adds one; the intended comparison is between operations for general i.

B - Incorrect: "i=i+1; E. ++i;" does not match the rule or result derived here. i=i+i doubles i rather than adding a fixed one. i+=1, i=i+1, i++ and the embedded ++i all increment by one. At the special value i=1 doubling also adds one; the intended comparison is between operations for general i.

C - Correct: "i=i+i;" matches the derived result. i=i+i doubles i rather than adding a fixed one. i+=1, i=i+1, i++ and the embedded ++i all increment by one. At the special value i=1 doubling also adds one; the intended comparison is between operations for general i.

D - Incorrect: "i++;" does not match the rule or result derived here. i=i+i doubles i rather than adding a fixed one. i+=1, i=i+1, i++ and the embedded ++i all increment by one. At the special value i=1 doubling also adds one; the intended comparison is between operations for general i.

Primary note proof

Study-note section CP08

> x+=y, x-=y, x*=y, x/=y and x%=y update x using the operation and conversion to its type. For simple variables, trace them as x=x op y. Starting p=q=n=8, p+=n gives 16 and p/=q gives 2. Starting x=10, x+=5 then x/=2 gives 7. x=x+x doubles; it does not generally add one.

Study-note section CP09

> Prefix ++x changes x and yields the new value; postfix x++ yields the previous value and also increments x.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science model exam · Question 6

University of Gondar model exam · Question 6

Question 28 | Q00336

Consider native C++ translation through assembly into an object file, stopping before linking. Which statement best describes the resulting object file’s instructions?

A. Machine-independent bytecode for a virtual machine

B. A complete linked executable

C. Both a and b

D. Native machine instructions in an object file

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: D. Native machine instructions in an object file

The specified native pipeline stops after the assembler produces an object file and before the linker combines objects and resolves external references. Its code is therefore native machine instructions in an object file. Machine-independent bytecode belongs to a different translation target, such as javac’s output, while a complete linked executable comes later in this C++ pipeline. Neither A nor B correctly names the specified result.

Why each choice is right or wrong

A - Incorrect: Machine-independent bytecode is a different target from the specified native object instructions.

B - Incorrect: The pipeline stops before linking produces a complete executable.

C - Incorrect: Both A and B describe different targets or later stages.

D - Correct: The assembler emits native machine instructions into an object file.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Study-note section CP04

> Java javac produces bytecode, which a JVM interprets or compiles at runtime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

Computer Science model exam · Question 67

University of Gondar model exam · Question 67

Question 29 | Q00455

A _____ provides us with named storage that our programs can manipulate.

A. data type

B. constants

C. operators

D. variable

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: variable

A variable is a symbolic name for a memory location holding data (p.5), so it supplies named manipulable storage. A data type classifies the value, constants have fixed values, and operators specify manipulations rather than storage. Module support: PDF p. 5.

Why each choice is right or wrong

A - Incorrect: "data type" does not match the rule or result derived here. A variable is a symbolic name for a memory location holding data (p.5), so it supplies named manipulable storage. A data type classifies the value, constants have fixed values, and operators specify manipulations rather than storage.

B - Incorrect: "constants" does not match the rule or result derived here. A variable is a symbolic name for a memory location holding data (p.5), so it supplies named manipulable storage. A data type classifies the value, constants have fixed values, and operators specify manipulations rather than storage.

C - Incorrect: "operators" does not match the rule or result derived here. A variable is a symbolic name for a memory location holding data (p.5), so it supplies named manipulable storage. A data type classifies the value, constants have fixed values, and operators specify manipulations rather than storage.

D - Correct: "variable" matches the derived result. A variable is a symbolic name for a memory location holding data (p.5), so it supplies named manipulable storage. A data type classifies the value, constants have fixed values, and operators specify manipulations rather than storage.

Primary note proof

Study-note section CP06

> Write type name; or type name = value; for ordinary variables: int age = 20;. Declaration introduces a name and type; initialization gives its initial value; later assignment replaces a value. a=b copies a scalar value, so later changes to b do not change a. A pointer or Java object reference instead copies an address/reference.

Study-note section CP25

> int x=10; int* p=&x; stores x’s address in p.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 24-33

> 5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Question sources

Object-Oriented Programming practice questions · Question 25

Question 30 | Q00464

The && and || operators

A. Compare two boolean values

B. Compare two numeric values

C. Combine two boolean values

D. Combine two numeric values

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Combine two boolean values

The logical operators combine truth-valued operands and produce 1 or 0 (p.7). Relational operators perform comparisons; arithmetic operators combine numeric quantities arithmetically. Numeric operands may be interpreted as truth values, but the logical role asked here is combination of boolean values. Module support: PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "Compare two boolean values" does not match the rule or result derived here. The logical operators combine truth-valued operands and produce 1 or 0 (p.7). Relational operators perform comparisons; arithmetic operators combine numeric quantities arithmetically. Numeric operands may be interpreted as truth values, but the logical role asked here is combination of boolean values.

B - Incorrect: "Compare two numeric values" does not match the rule or result derived here. The logical operators combine truth-valued operands and produce 1 or 0 (p.7). Relational operators perform comparisons; arithmetic operators combine numeric quantities arithmetically. Numeric operands may be interpreted as truth values, but the logical role asked here is combination of boolean values.

C - Correct: "Combine two boolean values" matches the derived result. The logical operators combine truth-valued operands and produce 1 or 0 (p.7). Relational operators perform comparisons; arithmetic operators combine numeric quantities arithmetically. Numeric operands may be interpreted as truth values, but the logical role asked here is combination of boolean values.

D - Incorrect: "Combine two numeric values" does not match the rule or result derived here. The logical operators combine truth-valued operands and produce 1 or 0 (p.7). Relational operators perform comparisons; arithmetic operators combine numeric quantities arithmetically. Numeric operands may be interpreted as truth values, but the logical role asked here is combination of boolean values.

Primary note proof

Study-note section CP10

> Relational and equality operators test values; && and || combine truth values; ! negates. In C++, zero converts to false and a nonzero number to true. Streams display bool as 0 or 1 unless boolalpha is set. With a=3,b=5, cout<<(a==b) prints 0.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Object-Oriented Programming practice questions · Question 34

Question 31 | Q00470

Which of these have highest precedence?

A. ()

B. ++

C. *

D. >>

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: Parentheses ().

CP10 places grouping/postfix at the top of its precedence ladder. Parentheses explicitly group an expression before surrounding operations are applied. Increment, multiplication or dereference, and right shift do not override that grouping. The precise role of * or ++ in an expression does not make it outrank explicit parentheses.

Why each choice is right or wrong

A - Correct: Parentheses explicitly group expressions and occupy the highest grouping level in the note.

B - Incorrect: Increment does not override parentheses; prefix increment is unary and postfix increment follows grouping rules.

C - Incorrect: Multiplication is below grouping; unary dereference also does not override parenthesized grouping.

D - Incorrect: Right shift ranks below both grouping and arithmetic operations in the stated ladder.

Primary note proof

Study-note section CP10

> && evaluates its right operand only if the left is true; || only if the left is false. Thus false && dangerousCall() skips the call. Bitwise & operates on bits and does not short circuit. With x=1,y=0,z=4, (x&y)||++z evaluates ++z, leaving z=5 and the Boolean result 1. The precedence ladder is grouping/postfix, unary, multiplicative, additive, shifts, relations, equality, bitwise &, ^, |, &&, ||, conditional, assignment, comma.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Object-Oriented Programming practice questions · Question 40

Question 32 | Q00479

A _____ statement allows us to execute a statement or group of statements multiple times.

A. array

B. loop

C. function

D. exception

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: loop

Loop statements specify repeated computations (p.9), and iteration repeats a statement or block (p.11). Arrays group data, functions modularize code, and exceptions are not the module definition of repetition. Module support: PDF p. 9, PDF p. 11.

Why each choice is right or wrong

A - Incorrect: "array" does not match the rule or result derived here. Loop statements specify repeated computations (p.9), and iteration repeats a statement or block (p.11). Arrays group data, functions modularize code, and exceptions are not the module definition of repetition.

B - Correct: "loop" matches the derived result. Loop statements specify repeated computations (p.9), and iteration repeats a statement or block (p.11). Arrays group data, functions modularize code, and exceptions are not the module definition of repetition.

C - Incorrect: "function" does not match the rule or result derived here. Loop statements specify repeated computations (p.9), and iteration repeats a statement or block (p.11). Arrays group data, functions modularize code, and exceptions are not the module definition of repetition.

D - Incorrect: "exception" does not match the rule or result derived here. Loop statements specify repeated computations (p.9), and iteration repeats a statement or block (p.11). Arrays group data, functions modularize code, and exceptions are not the module definition of repetition.

Primary note proof

Study-note section CP14

> while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Study-note section CP32

> Exceptions transfer control to a matching handler for exceptional conditions; they do not make arbitrary undefined behavior recoverable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 9 | lines 2-8

> Control Statements Control statements or flow control determines what is executed during a run and what is not, therefore affecting the overall outcome of the program. C++ provides different forms of statements for different purposes. Declaration statements are used for defining variables. Assignment-like statements are used for simple, algebraic computations. Branching statements are used for specifying alternate paths of execution, depending on the outcome of a

logical condition. Loop statements are used for specifying computations, which need to be repeated until a certain logical condition is satisfied

Computer Programming module | PDF page 11 | lines 5-13

Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Question sources

Object-Oriented Programming practice questions · Question 49

Question 33 | Q00482

A for loop is useful when you know how many times a task is to be repeated.

A. TRUE

B. FALSE

C. Can be true or false

D. Can not say

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: TRUE

A for loop has initialization, a condition and an update field (p.12). These directly express a known repetition count using a counter, so the stated usefulness is true; it does not claim that all for loops must have predetermined counts. Module support: PDF p. 12.

Why each choice is right or wrong

A - Correct: "TRUE" matches the derived result. A for loop has initialization, a condition and an update field (p.12). These directly express a known repetition count using a counter, so the stated usefulness is true; it does not claim that all for loops must have predetermined counts.

B - Incorrect: "FALSE" does not match the rule or result derived here. A for loop has initialization, a condition and an update field (p.12). These directly express a known repetition count using a counter, so the stated usefulness is true; it does not claim that all for loops must have predetermined counts.

C - Incorrect: "Can be true or false" does not match the rule or result derived here. A for loop has initialization, a condition and an update field (p.12). These directly express a known repetition

count using a counter, so the stated usefulness is true; it does not claim that all for loops must have predetermined counts.

D - Incorrect: "Can not say" does not match the rule or result derived here. A for loop has initialization, a condition and an update field (p.12). These directly express a known repetition count using a counter, so the stated usefulness is true; it does not claim that all for loops must have predetermined counts.

Primary note proof

Study-note section CP14

> for(initialization; condition; update) executes initialization once, then condition → body → update → condition. A missing condition in for(;;) acts as true. A for loop is convenient for a known count but can also be condition controlled. To print indices 0 through n-1, use i=0; i<n; ++i. Starting i=0 with i<5 and i+=2 prints 0 2 4.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 19-28

> The for loop. Its format is: for (initialization; condition; increase) statement; 1, initialization is executed. Generally, it is a initial value setting for a counter variable. This is executed only once 2, condition is checked, if it is true the loop continues, otherwise the loop finishes and statement is skipped. 3, statement is executed. As usual, it can be either a single instruction or a block of instructions enclosed within curly brackets { }. 4, finally, whatever is specified in the increase field is executed and the loop gets back to step 2.

Question sources

Object-Oriented Programming practice questions · Question 52

Question 34 | Q00627

Which one of the following are translate the whole source program into object code.

A. Compiler

B. Interprater

C. Linker

D. All

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: Compiler

The module says compilers translate the entire source before execution and produce object code (pp.3-4). Interpreters translate one line at a time. Linking combines existing object files rather than translating the source, so All cannot hold. Module support: PDF p. 4, PDF p. 3.

Why each choice is right or wrong

A - Correct: "Compiler" matches the derived result. The module says compilers translate the entire source before execution and produce object code (pp.3-4). Interpreters translate one line at a time. Linking combines existing object files rather than translating the source, so All cannot hold.

B - Incorrect: "Interprater" does not match the rule or result derived here. The module says compilers translate the entire source before execution and produce object code (pp.3-4). Interpreters translate one line at a time. Linking combines existing object files rather than translating the source, so All cannot hold.

C - Incorrect: "Linker" does not match the rule or result derived here. The module says compilers translate the entire source before execution and produce object code (pp.3-4). Interpreters translate one line at a time. Linking combines existing object files rather than translating the source, so All cannot hold.

D - Incorrect: "All" does not match the rule or result derived here. The module says compilers translate the entire source before execution and produce object code (pp.3-4). Interpreters translate one line at a time. Linking combines existing object files rather than translating the source, so All cannot hold.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Study-note section CP04

> GCC documents the first four stages explicitly. A compiler need not emit a complete executable or even native machine code. Java javac produces bytecode, which a JVM interprets or compiles at runtime. Interpreters execute a program representation; “one source line at a time” is a beginner approximation. Compilation uses multiple passes, so “whole program in one step” is not literally universal.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started

that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Computer Programming module | PDF page 3 | lines 8-17

Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Computer Science model exam · 2015 · Question 96

Question 35 | Q00660

In C++ programming, a variable which stores a reference to another variable is called a____

A. Pointer

B. Constructor

C. Asterisks

D. Structure

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: Pointer

The module defines a pointer as a variable holding the memory address of another variable (p.26). Here reference is informal wording for an address, not the distinct C++ reference type. Constructor, the asterisk symbol and structure do not name that variable category. Module support: PDF p. 26.

Why each choice is right or wrong

A - Correct: "Pointer" matches the derived result. The module defines a pointer as a variable holding the memory address of another variable (p.26). Here reference is informal wording for an address, not the distinct C++ reference type. Constructor, the asterisk symbol and structure do not name that variable category.

B - Incorrect: "Constructor" does not match the rule or result derived here. The module defines a pointer as a variable holding the memory address of another variable (p.26). Here reference is

informal wording for an address, not the distinct C++ reference type. Constructor, the asterisk symbol and structure do not name that variable category.

C - Incorrect: "Asterisks" does not match the rule or result derived here. The module defines a pointer as a variable holding the memory address of another variable (p.26). Here reference is informal wording for an address, not the distinct C++ reference type. Constructor, the asterisk symbol and structure do not name that variable category.

D - Incorrect: "Structure" does not match the rule or result derived here. The module defines a pointer as a variable holding the memory address of another variable (p.26). Here reference is informal wording for an address, not the distinct C++ reference type. Constructor, the asterisk symbol and structure do not name that variable category.

Primary note proof

Study-note section CP25

int x=10; int* p=&x; stores x's address in p. *p reads or writes x; &p is the address of the pointer variable itself. With p=&a,q=&b, p=q redirects p to b and leaves a and b unchanged. By contrast *p=*q copies the pointed-to value. Never dereference an uninitialized, null or dangling pointer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 26 | lines 2-22

Pointer - A pointer is a variable that holds a memory address of another variable. - - Its content is a memory address - A pointer is simply the address of a memory location and provides an indirect way of accessing data in memory. - It is defined to „point to" data of a specific type. - Because a pointer variable holds the address of another piece of data, it "points to" the data Reference operator (&) and Dereference operator (*) &: read as address of - Use address operator & to get address of a variable *: read as value pointed by Declaring variables of pointer types Syntax: type *variable_name; For example: int *ptr1; //pointertoanint double*number; char*ptr2; //pointertoachar The value of a pointer variable is the address to which it points. For example, given the definitions int num; wecanwrite: ptr1=#

Question sources

Computer Science model exam 2 · Question 1

Question 36 | Q00661

A program that translates a high-level language like C++ to a machine language is called a

A. Disassembler

B. Coder

C. Compiler

D. Assembler

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Compiler

A compiler translates the high-level source program before execution (pp.3-4). A linker combines object files and an assembler is not named here as the high-level translator; disassembler/coder do not match the stated compilation role. Module support: PDF p. 3, PDF p. 4.

Why each choice is right or wrong

A - Incorrect: "Disassembler" does not match the rule or result derived here. A compiler translates the high-level source program before execution (pp.3-4). A linker combines object files and an assembler is not named here as the high-level translator; disassembler/coder do not match the stated compilation role.

B - Incorrect: "Coder" does not match the rule or result derived here. A compiler translates the high-level source program before execution (pp.3-4). A linker combines object files and an assembler is not named here as the high-level translator; disassembler/coder do not match the stated compilation role.

C - Correct: "Compiler" matches the derived result. A compiler translates the high-level source program before execution (pp.3-4). A linker combines object files and an assembler is not named here as the high-level translator; disassembler/coder do not match the stated compilation role.

D - Incorrect: "Assembler" does not match the rule or result derived here. A compiler translates the high-level source program before execution (pp.3-4). A linker combines object files and an assembler is not named here as the high-level translator; disassembler/coder do not match the stated compilation role.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

Computer Science model exam 2 · Question 2

Question 37 | Q00662

Given int A=0, B=10, x=1, y=2;, which C++ Boolean expression evaluates to false?

A. (B == 10) || (x < y)

B. (A > 20) || (B >5)

C. (A > 10) && (x < y)

D. (A == 0) && (B < 20)

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: C. (A > 10) && (x < y)

Evaluate each comparison using the declared integers and apply short-circuit logic. B==10 is true, so the first disjunction is true. Although A>20 is false, B>5 makes the second disjunction true. In C, A>10 is false, so && yields false without needing its right operand. Both comparisons in D are true. Every offered expression is well formed under the stated declarations.

Why each choice is right or wrong

A - Incorrect: B==10 is true, so this disjunction is true.

B - Incorrect: B>5 is true, so this disjunction is true.

C - Correct: A>10 is false, making the conjunction false.

D - Incorrect: A==0 and B<20 are both true.

Primary note proof

Study-note section CP10

> Relational and equality operators test values; && and || combine truth values; ! negates. In C++, zero converts to false and a nonzero number to true. Streams display bool as 0 or 1 unless boolalpha is set. With a=3,b=5, cout<<(a==b) prints 0.

Study-note section CP10

> && evaluates its right operand only if the left is true; || only if the left is false.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science model exam 2 · Question 3

Question 38 | Q00663

Which one of the following format is write about functions in structural Programming:

A. Type name { parameter1, parameter2, ...} (statement)

B. Name type (parameter1, parameter2, ...) { statement }

C. Type name (parameter1, parameter2, ...) {statement }

D. Name type { parameter1, parameter2, ...} (statement)

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Type name (parameter1, parameter2, ...) {statement }

The module function-definition pattern is Function_type function_name(Parameter List) followed by a braced function body (p.15). C preserves that order and delimiters. A/D interchange braces and parentheses, while B puts name before type. Module support: PDF p. 15.

Why each choice is right or wrong

A - Incorrect: "Type name { parameter1, parameter2, ...} (statement)" does not match the rule or result derived here. The module function-definition pattern is Function_type function_name(Parameter List) followed by a braced function body (p.15). C preserves that order and delimiters. A/D interchange braces and parentheses, while B puts name before type.

B - Incorrect: "Name type (parameter1, parameter2, ...) { statement }" does not match the rule or result derived here. The module function-definition pattern is Function_type function_name(Parameter List) followed by a braced function body (p.15). C preserves that order and delimiters. A/D interchange braces and parentheses, while B puts name before type.

C - Correct: "Type name (parameter1, parameter2, ...) {statement }" matches the derived result. The module function-definition pattern is Function_type function_name(Parameter List) followed by a braced function body (p.15). C preserves that order and delimiters. A/D interchange braces and parentheses, while B puts name before type.

D - Incorrect: "Name type { parameter1, parameter2, ...} (statement)" does not match the rule or result derived here. The module function-definition pattern is Function_type function_name(Parameter List) followed by a braced function body (p.15). C preserves that order and delimiters. A/D interchange braces and parentheses, while B puts name before type.

Primary note proof

Study-note section CP17

A function groups a named task. C++ definition: int add(int a,int b){return a+b;}. Call it with add(x,y);, without parameter type declarations at the call site. Parameters are names in the definition; arguments are supplied expressions or values. A void function produces no return value but can have side effects and return; to exit early.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 29-46

• User-defined - Function that been created by the user. - These functions need to be declared and defined by the user. Declaration of function Function defines using the following syntax: Function_type function_name (Parameter List) { ...function body } Example: void Addition(int num1,int num2) #include<iostream> using namespace std; void Addition(int num1,int num2) { int sum=num1+num2; cout<<"The sum is:"<<sum; } int main() { int x=10,y=20; Addition(x,y);

Question sources

Computer Science model exam 2 · Question 4

Question 39 | Q00664

Which of the following is a valid function call (assuming the function add exists & have two parameters)?

A. add;

B. add(a,b);

C. add a,b;

D. add(int a, int b);

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer B: add(a,b);

The example Addition(x,y) on p.15 calls a function using its name and parenthesized argument expressions. add(a,b); has that form. A does not call it, C lacks parentheses, and D inserts declarations where argument expressions are required. Module support: PDF p. 15.

Why each choice is right or wrong

A - Incorrect: "add;" does not match the rule or result derived here. The example Addition(x,y) on p.15 calls a function using its name and parenthesized argument expressions. add(a,b); has that form. A does not call it, C lacks parentheses, and D inserts declarations where argument expressions are required.

B - Correct: "add(a,b);" matches the derived result. The example Addition(x,y) on p.15 calls a function using its name and parenthesized argument expressions. add(a,b); has that form. A does not call it, C lacks parentheses, and D inserts declarations where argument expressions are required.

C - Incorrect: "add a,b;" does not match the rule or result derived here. The example Addition(x,y) on p.15 calls a function using its name and parenthesized argument expressions. add(a,b); has that form. A does not call it, C lacks parentheses, and D inserts declarations where argument expressions are required.

D - Incorrect: "add(int a, int b);" does not match the rule or result derived here. The example Addition(x,y) on p.15 calls a function using its name and parenthesized argument expressions. add(a,b); has that form. A does not call it, C lacks parentheses, and D inserts declarations where argument expressions are required.

Primary note proof

Study-note section CP17

> A function groups a named task. C++ definition: int add(int a,int b){return a+b;}. Call it with add(x,y);, without parameter type declarations at the call site. Parameters are names in the definition; arguments are supplied expressions or values. A void function produces no return value but can have side effects and return; to exit early.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 29-46

> • User-defined - Function that been created by the user. - These functions need to be declared and defined by the user. Declaration of function Function defines using the following syntax: Function_type function_name (Parameter List) { ...function body } Example: void Addition(int num1,int num2) #include<iostream> using namespace std; void Addition(int num1,int num2) { int sum=num1+num2; cout<<"The sum is:"<<sum; } int main() { int x=10,y=20; Addition(x,y);

Question sources

Computer Science model exam 2 · Question 5

Question 40 | Q00665

A function is said to be recursive function_____?

A. If the function calls another function2 & Function3

B. If the main function calls many functions

C. If the main function calls another one function

D. If the function calls itself

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: If the function calls itself

A recursive function calls itself (p.18). Calling several other functions, or being called by main, does not itself constitute recursion. Module support: PDF p. 18.

Why each choice is right or wrong

A - Incorrect: "If the function calls another function2 & Function3" does not match the rule or result derived here. A recursive function calls itself (p.18). Calling several other functions, or being called by main, does not itself constitute recursion.

B - Incorrect: "If the main function calls many functions" does not match the rule or result derived here. A recursive function calls itself (p.18). Calling several other functions, or being called by main, does not itself constitute recursion.

C - Incorrect: "If the main function calls another one function" does not match the rule or result derived here. A recursive function calls itself (p.18). Calling several other functions, or being called by main, does not itself constitute recursion.

D - Correct: "If the function calls itself" matches the derived result. A recursive function calls itself (p.18). Calling several other functions, or being called by main, does not itself constitute recursion.

Primary note proof

Study-note section CP21

> A recursive function calls itself directly or indirectly. It needs a base case and progress toward it for all supported inputs. factorial(0)=1 and factorial(n)=n*factorial(n-1) for positive n. factorial(3) expands to 3*2*1*1=6. Negative inputs require rejection or a separate definition; they do not reach zero under n-1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 18 | lines 23-28

> Recursive Functions - Functions that call themselves - A function that is defined in terms of itself is called recursive - There must be a base case e.g. 0! = 1 - Slowly converges towards base case - For example, Recursive functions N! = N * (N-1)!

Question sources

Computer Science model exam 2 · Question 6

Question 41 | Q00666

What will be the return type of a function, if the function does not have a return value?

A. Int

B. Float

C. Double

D. Void

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Void

The module illustrates void Addition(...) whose body prints a result without returning a value (p.15). This matches void for no returned value; int, float and double denote value-bearing return types. Capitalization in the options is treated as prose. Module support: PDF p. 15.

Why each choice is right or wrong

A - Incorrect: "Int" does not match the rule or result derived here. The module illustrates void Addition(...) whose body prints a result without returning a value (p.15). This matches void for no returned value; int, float and double denote value-bearing return types. Capitalization in the options is treated as prose.

B - Incorrect: "Float" does not match the rule or result derived here. The module illustrates void Addition(...) whose body prints a result without returning a value (p.15). This matches void for no returned value; int, float and double denote value-bearing return types. Capitalization in the options is treated as prose.

C - Incorrect: "Double" does not match the rule or result derived here. The module illustrates void Addition(...) whose body prints a result without returning a value (p.15). This matches void for no returned value; int, float and double denote value-bearing return types. Capitalization in the options is treated as prose.

D - Correct: "Void" matches the derived result. The module illustrates void Addition(...) whose body prints a result without returning a value (p.15). This matches void for no returned value; int, float and double denote value-bearing return types. Capitalization in the options is treated as prose.

Primary note proof

Study-note section CP17

A function groups a named task. C++ definition: int add(int a,int b){return a+b;}. Call it with add(x,y);, without parameter type declarations at the call site. Parameters are names in the definition; arguments are supplied expressions or values. A void function produces no return value but can have side effects and return; to exit early.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 29-46

• User-defined - Function that been created by the user. - These functions need to be declared and defined by the user. Declaration of function Function defines using the following syntax: Function_type function_name (Parameter List) { ...function body } Example: void Addition(int num1,int num2) #include<iostream> using namespace std; void Addition(int num1,int num2) { int sum=num1+num2; cout<<"The sum is:"<<sum; } int main() { int x=10,y=20; Addition(x,y);

Question sources

Computer Science model exam 2 · Question 7

Question 42 | Q00667

____is a header file which permits writing of data to a file.

A. ofstream

B. ifstream

C. istream

D. fstream

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: fstream

The module explicitly instructs including the <fstream> header to work with files (p.40). ofstream is the class for writing, ifstream the reading class, and istream is not the named file header. The question asks for the header, so fstream is the applicable choice. Module support: PDF p. 40.

Why each choice is right or wrong

A - Incorrect: "ofstream" does not match the rule or result derived here. The module explicitly instructs including the <fstream> header to work with files (p.40). ofstream is the class for writing, ifstream the reading class, and istream is not the named file header. The question asks for the header, so fstream is the applicable choice.

B - Incorrect: "ifstream" does not match the rule or result derived here. The module explicitly instructs including the <fstream> header to work with files (p.40). ofstream is the class for writing, ifstream the reading class, and istream is not the named file header. The question asks for the header, so fstream is the applicable choice.

C - Incorrect: "istream" does not match the rule or result derived here. The module explicitly instructs including the <fstream> header to work with files (p.40). ofstream is the class for writing, ifstream the reading class, and istream is not the named file header. The question asks for the header, so fstream is the applicable choice.

D - Correct: "fstream" matches the derived result. The module explicitly instructs including the <fstream> header to work with files (p.40). ofstream is the class for writing, ifstream the reading class, and istream is not the named file header. The question asks for the header, so fstream is the applicable choice.

Primary note proof

Study-note section CP29

> Include <fstream>. ofstream writes, ifstream reads, and fstream can do both. The constructor's default ofstream mode argument is ios::out; its ordinary effect is to create/truncate the file. Distinguish the default mode argument from effective out|trunc behavior. ios::app places each write at the end; ios::ate initially seeks to the end; ios::binary requests binary mode.

Study-note section CP29

> "ofstream" is a class, whereas <fstream> is the header.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 40 | lines 16-22

> File and File Management Files are used to store data in storage devices permanently. The fstream library allows us to work with files. To use the fstream library, include both the standard <iostream> AND the <fstream> header file: Input/Output with files • ofstream: Stream class to write on files • ifstream: Stream class to read from files

Question sources

Computer Science model exam 2 · Question 8

Question 43 | Q00668

Which statements about C++ structures are true? Select all that apply. In C, initialization refers to defining a structure object with initial member values.

A. A structure must contain only homogeneous members.

B. The named individual components of a structure are called members.

C. A structure object can be initialized when it is defined.

D. All are correct

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B, C: B. The named individual components of a structure are called members.; C. A structure object can be initialized when it is defined.

A structure groups named members and can combine different member types, so it need not be homogeneous. Those named components are called members. Defining an object with an aggregate initializer supplies initial values in member declaration order, making the initialization statement true. Since A wrongly requires homogeneity, the aggregate claim that all statements are correct is false even though B and C hold.

Why each choice is right or wrong

A - Incorrect: Members can have different types; homogeneity is not required.

B - Correct: The named individual components of a structure are its members.

C - Correct: A structure object can be initialized when the object is defined.

D - Incorrect: A is false, so the aggregate all-correct claim fails.

Primary note proof

Study-note section CP27

> A struct groups named members, possibly of different types. struct House{double price; int rooms;}; defines a type; House h{150000,3}; creates an object. h.rooms uses a dot. House* p=&h; p->rooms is equivalent to (*p).rooms. The pointer must refer to a live object before member access. A type definition alone does not create an ordinary instance.

Study-note section CP27

> Aggregate initialization assigns members in declaration order.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 35 | lines 13-17

> Accessing members of a structure variable A structure member can be accessed using dot operator'.' notation syntax: structvariable.member; eg. st.id=4;

Computer Programming module | PDF page 35 | lines 37-44

> Initialize value #include<iostream> using namespace std; struct student{ string name; int age; double score; }st={"Abebe",19,24};

Question sources

Computer Science model exam 2 · Question 9

Question 44 | Q00669

Which scope category describes a variable declared outside functions, usable where its declaration is visible and potentially hidden by a local variable of the same name?

A. Global Variable

B. Limited Variables

C. Local Variable

D. Partial Variable

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: A. Global Variable

A namespace or global variable is declared outside a function. Its declaration must be visible at the point of use, so global does not mean automatically accessible in every file or scope. A local name can shadow it, and C++ scope resolution can select a hidden global name. This makes Global Variable the appropriate category while avoiding the earlier overstatement about unrestricted program-wide accessibility.

Why each choice is right or wrong

A - Correct: Global variables have the stated outside-function scope, subject to declaration visibility.

B - Incorrect: Limited Variables is not the scope category specified by the declarations.

C - Incorrect: A local variable is declared within a function or block.

D - Incorrect: Partial Variable is not the stated C++ scope category.

Primary note proof

Study-note section CP20

> Scope is where a name can be used; lifetime is when its object exists. A local name is restricted to its block; an outer name may be shadowed. A namespace/global variable is available where its declaration is visible, not magically in every file. C++ ::name selects a global name hidden by a local one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 17 | lines 14-29

> Scope of variables Scope of variables defined as the extent of the program code within which the variable can be accessed or declared or worked with mainly two types of variable scopes: local variable: - Variables defined within a function or block are said to be local to those functions - Declared inside a block{} Global Variable - variable that can be accessed from any part of the program. - They are available throughout the life time of a program. - They are declared at the

top of the program outside all of the functions or blocks. NB: - Usually when two variables with same name are defined then the compiler produces a compile time error. But if the variables are defined in different scopes, then the compiler allows it. - Whenever there is a local variable defined with same name as that of a global variable then the compiler will give precedence to the local variable

Question sources

Computer Science model exam 2 · Question 10

Question 45 | Q00906

What is the output of the following C++ fragment code? int a = 6, b = 8; int x = 2, y = 4; int c = (x > y ? (a--, x) : (b--, y)); cout << "a=" << a; cout << " b=" << b; cout << " c=" << c;

A. a=5 b=7 c=2

B. a=5 b=8 c=4

C. a=6 b=8 c=4

D. a=6 b=7 c=4

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: a=6 b=7 c=4

x>y is 2>4, false, so only the third operand (b--,y) is evaluated. It decrements b from 8 to 7 and yields y=4; a remains 6. Thus a=6 b=7 c=4. A/B incorrectly decrement a; C omits the selected decrement. Module support: PDF p. 8, PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "a=5 b=7 c=2" does not match the rule or result derived here. x>y is 2>4, false, so only the third operand (b--,y) is evaluated. It decrements b from 8 to 7 and yields y=4; a remains 6. Thus a=6 b=7 c=4. A/B incorrectly decrement a; C omits the selected decrement.

B - Incorrect: "a=5 b=8 c=4" does not match the rule or result derived here. x>y is 2>4, false, so only the third operand (b--,y) is evaluated. It decrements b from 8 to 7 and yields y=4; a remains 6. Thus a=6 b=7 c=4. A/B incorrectly decrement a; C omits the selected decrement.

C - Incorrect: "a=6 b=8 c=4" does not match the rule or result derived here. x>y is 2>4, false, so only the third operand (b--,y) is evaluated. It decrements b from 8 to 7 and yields y=4; a remains 6. Thus a=6 b=7 c=4. A/B incorrectly decrement a; C omits the selected decrement.

D - Correct: "a=6 b=7 c=4" matches the derived result. x>y is 2>4, false, so only the third operand (b--,y) is evaluated. It decrements b from 8 to 7 and yields y=4; a remains 6. Thus a=6 b=7 c=4. A/B incorrectly decrement a; C omits the selected decrement.

Primary note proof

Study-note section CP11

> condition ? yes : no evaluates the condition and exactly one alternative. The comma operator evaluates its left expression, then its right, and yields the right value. Do not confuse it with commas separating function arguments.

Study-note section CP09

> Prefix ++x changes x and yields the new value; postfix x++ yields the previous value and also increments x.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 8 | lines 2-6

> Ternary /Conditional operator (? :) A ternary operator evaluates the test condition and executes a block of code based on the result of the condition. Syntax: Condition? Expression 1: Expression2 If the condition is true expression 1 will be evaluated otherwise Expression 2 will evaluated.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science Exit Exam · 2018 · Question 19

MOE Computer Science Exit Exam · Question 25-2

National Computer Science Exit Exam · Question 23

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 77

Question 46 | Q00936

Which of the following is NOT property of an algorithm?

A. Finiteness

B. Definiteness

C. Correctness

D. Platform dependence

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Platform dependence

The module properties include correctness, unambiguity and finiteness (p.2). Platform dependence is not a property demanded of an abstract algorithm. Definiteness corresponds to clearly defined unambiguous steps; D is the extraneous platform restriction. Module support: PDF p. 2.

Why each choice is right or wrong

A - Incorrect: "Finiteness" does not match the rule or result derived here. The module properties include correctness, unambiguity and finiteness (p.2). Platform dependence is not a property demanded of an abstract algorithm. Definiteness corresponds to clearly defined unambiguous steps; D is the extraneous platform restriction.

B - Incorrect: "Definiteness" does not match the rule or result derived here. The module properties include correctness, unambiguity and finiteness (p.2). Platform dependence is not a property demanded of an abstract algorithm. Definiteness corresponds to clearly defined unambiguous steps; D is the extraneous platform restriction.

C - Incorrect: "Correctness" does not match the rule or result derived here. The module properties include correctness, unambiguity and finiteness (p.2). Platform dependence is not a property demanded of an abstract algorithm. Definiteness corresponds to clearly defined unambiguous steps; D is the extraneous platform restriction.

D - Correct: "Platform dependence" matches the derived result. The module properties include correctness, unambiguity and finiteness (p.2). Platform dependence is not a property demanded of an abstract algorithm. Definiteness corresponds to clearly defined unambiguous steps; D is the extraneous platform restriction.

Primary note proof

Study-note section CP02

> An algorithm gives precise, executable steps and terminates for every legal input. Definiteness means each step has an unambiguous meaning; effectiveness means a step can actually be performed. Inputs may be absent, but a useful algorithm specifies a result. Correctness means the result meets the specification. Efficiency is desirable and measurable, but an inefficient

terminating algorithm is still an algorithm. Efficiency is also a desirable property of an algorithm.

Study-note section CP01

Solve one small case manually, choose an algorithm, express it as pseudocode or a flowchart, implement, test and maintain.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 19-26

Algorithm is an ordered set of unambiguous, executable steps, defining a terminating process. Important Properties of Algorithms - Correct: always returns the desired output for all legal instances of the problem. - Unambiguous: Each step must be clearly defined, having only one interpretation. - Precise: simple but efficient - Efficient: can be measured in terms of Time and Space - Finiteness: the execution of the algorithm must terminate in a finite amount of time for each possible set of input data/must have an end.

Question sources

Computer Science Exit Exam · 2018 · Question 49

Question 47 | Q00938

Assume that you are computer science expert and you need to develop large programs within short period of time. Finally, you have planned to provide portable program to the concerned body. Which type of programming language is preferable for the above scenario?

A. Low level

B. Machine level

C. .
High level

D. Assembly

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: .
High level

The decisive requirement is portability: the module says high-level languages are machine-independent and run on a variety of computers (p.3). Machine and assembly are low-level and machine-dependent. Thus C meets portability; the module does not quantify development time. Module support: PDF p. 3.

Why each choice is right or wrong

A - Incorrect: "Low level" does not match the rule or result derived here. The decisive requirement is portability: the module says high-level languages are machine-independent and run on a variety of computers (p.3). Machine and assembly are low-level and machine-dependent. Thus C meets portability; the module does not quantify development time.

B - Incorrect: "Machine level" does not match the rule or result derived here. The decisive requirement is portability: the module says high-level languages are machine-independent and run on a variety of computers (p.3). Machine and assembly are low-level and machine-dependent. Thus C meets portability; the module does not quantify development time.

C - Correct: ".
High level" matches the derived result. The decisive requirement is portability: the module says high-level languages are machine-independent and run on a variety of computers (p.3). Machine and assembly are low-level and machine-dependent. Thus C meets portability; the module does not quantify development time.

D - Incorrect: "Assembly" does not match the rule or result derived here. The decisive requirement is portability: the module says high-level languages are machine-independent and run on a variety of computers (p.3). Machine and assembly are low-level and machine-dependent. Thus C meets portability; the module does not quantify development time.

Primary note proof

Study-note section CP03

> Machine and assembly code depend on an instruction set. High-level source is easier to read and often portable after recompilation, provided it avoids platform assumptions. People can write both assembly and high-level programs. Python, Perl and PHP are commonly called scripting languages; C++ is conventionally compiled. These labels describe common use, not an absolute technical prohibition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Computer Science Exit Exam · 2018 · Question 51

Question 48 | Q00966

Which identifier is invalid under the basic ASCII C++ identifier rules? Treat the stray leading dot before jan2025 as an OCR artifact.

A. delete

B. for_cpp

C. jan2025

D. Hello

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer A: A. delete

An ordinary identifier starts with a letter or underscore and continues with letters, digits or underscores; it must also avoid keywords. delete fails because it is a C++ keyword. for_cpp is a single identifier distinct from the keyword for, while jan2025 and Hello follow the character rules. Removing the documented stray dot makes the intended identifier comparison explicit without changing the keyword concept.

Why each choice is right or wrong

A - Correct: delete is a C++ keyword and cannot be an ordinary identifier.

B - Incorrect: for_cpp contains valid identifier characters and is not the keyword for.

C - Incorrect: jan2025 starts with a letter and continues with allowed letters and digits.

D - Incorrect: Hello contains only letters and is not a keyword.

Primary note proof

Study-note section CP05

> In the basic ASCII identifier rules, start with a letter or underscore; continue with letters, digits or underscores. Do not use a keyword. C++ is case sensitive: exitExam and ExitExam differ; delete and float are keywords, but FLOAT is a legal identifier. variable_1234 is valid; 4thInput, T-3 and names containing spaces are not single identifiers. Avoid reserved underscore patterns in application code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 2-10

> 1. Keywords (reserved words) - Unique meaning within a C++ program and must not be used for any other purposes. E.g., if, namespace for, switch, continue, float, new, default, break Etc. 2. Identifiers - Name associated with a function or data object and used to refer to that function or data object. An identifier must: o Start with a letter or underscore o Consist only of letters, the digits 0-9, or the underscore symbol _ o Not be a reserved word

Question sources

Computer Science Exit Exam · 2018 · Question 79

Question 49 | Q00989

Which C++ declarations denote the same signed short integer type as short n;? Select all that apply.

A. int short n;

B. unsigned short n;

C. float short n;

D. short n;

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer A, D: A. int short n;; D. short n;

The ordinary short type is signed unless unsigned is explicitly requested. The note identifies short n, short int n and int short n as equivalent spellings. Consequently, A is equivalent despite its reordered specifiers, and D is equivalent because it repeats the original declaration. B changes signedness, while C combines incompatible floating and integer type specifiers rather than naming the requested type.

Why each choice is right or wrong

A - Correct: int short is an equivalent spelling of signed short.

B - Incorrect: unsigned short is a different type with different signedness.

C - Incorrect: float short is not a valid combination of these type specifiers.

D - Correct: This repeats the original signed short declaration.

Primary note proof

Study-note section CP06

> Use int for whole numbers, float/double for fractional approximations, char for a character and bool for logical values in C++. Type sizes depend on the implementation; do not import Java widths into C++. short n, short int n and int short n denote the same signed short type. const int n=5 and int const n=5 are equivalent; an ordinary const int needs initialization. typedef creates an alias, not a distinct new type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 24-33

> 5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established

when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Question sources

National Exit Exam practice questions · Question 3

Computer Science Exit Exam practice collection · Question 82

Question 50 | Q00990

Which one of the following is NOT a high-level language?

A. C++

B. Java

C. Python

D. Assembly

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: D. Assembly

Machine and assembly languages are low-level because their instructions are tied to an instruction set. C++ and Java are general high-level languages in the note's conventional classification, and Python is also a high-level language even though it is commonly used for scripting. Assembly is therefore the exception requested here. Classification by language level is separate from whether an implementation compiles or interprets a program.

Why each choice is right or wrong

A - Incorrect: C++ is a general high-level language, so it is not the requested exception.

B - Incorrect: Java is a general high-level language, so it is not the requested exception.

C - Incorrect: Python is high-level; scripting usage does not make it assembly or machine code.

D - Correct: Assembly is an instruction-set-dependent low-level language.

Primary note proof

Study-note section CP03

Machine instructions are conventionally first generation; symbolic assembly is second; general high-level languages such as C++ and Java are third. Database query and application generators such as SQL are conventionally fourth generation. Fifth generation is a loose historical category associated with declarative logic, constraints and AI/expert systems. Generations do not determine whether an implementation uses an interpreter.

Study-note section CP-S02

Python is a high-level programming language with built-in high-level data types. Calling a language a scripting language describes a common way it is used; it does not make that language machine code or assembly. Language level and execution style are different classifications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

National Exit Exam practice questions · Question 4

Computer Science Exit Exam practice collection · Question 83

Question 51 | Q00998

What does this C++ fragment print? Assume cout is available.
int x=15,y=2;
int z=x/y;
cout<<z;

A. 7.5

B. 7

C. 3

D. 12.5

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: B. 7

Both division operands are int, so their quotient is computed using integer division. Fifteen divided by two truncates toward zero to 7, which is stored in z and printed. The lowercase identifier matches the declared variable. A fractional quotient would require conversion before division, and none occurs here.

Why each choice is right or wrong

A - Incorrect: The fractional mathematical quotient is discarded by integer division.

B - Correct: The int quotient is 7 and is printed from z.

C - Incorrect: 3 is not the quotient of the given operands.

D - Incorrect: 12.5 follows neither integer division nor the mathematical quotient.

Primary note proof

Study-note section CP08

> Multiplication, division and remainder have higher precedence than addition and subtraction; equal-precedence arithmetic groups left to right. Integer division truncates toward zero. For integers, a == (a/b)*b + a%b when the division is valid. % requires integral operands in C++; (8.0*3)%5 is invalid. Division by zero and signed integer overflow cannot be given ordinary numeric outputs.

Study-note section CP34

> Printing Z when only z is declared refers to a different, undeclared identifier.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, | |) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 | | 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 6 | lines 4-15

> 6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

National Exit Exam practice questions · Question 12

Question 52 | Q01001

Which generation of programming languages stated for interpreted languages such as SQL?

A. 4th generation

B. 2nd generation

C. 3rd generation

D. All of the above

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: SQL is conventionally fourth generation.

CP03 explicitly classifies SQL with fourth-generation query languages and application generators. The question's description of interpreted languages is imprecise: interpretation is not what determines a language generation. Nevertheless, the named example SQL has a clear conventional classification, making fourth generation the supported answer.

Why each choice is right or wrong

A - Correct: Fourth generation is the note's explicit conventional classification for SQL and query languages.

B - Incorrect: Second generation refers to assembly; interpretation alone does not move SQL into this category.

C - Incorrect: Third generation describes general high-level languages, rather than SQL's stated query-language category.

D - Incorrect: SQL is not classified into all these generations merely because implementations can interpret it.

Primary note proof

Study-note section CP03

> Machine instructions are conventionally first generation; symbolic assembly is second; general high-level languages such as C++ and Java are third. Database query and application generators such as SQL are conventionally fourth generation. Fifth generation is a loose historical category associated with declarative logic, constraints and AI/expert systems. Generations do not determine whether an implementation uses an interpreter.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of

computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

National Exit Exam practice questions · Question 15

Computer Science Exit Exam practice collection · Question 85

Question 53 | Q01130

A variable name can only contain _____?

A. alphanumeric characters

B. underscores

C. Both A and B

D. None of the above

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Both A and B

The permitted identifier characters are letters, digits and underscore (p.5), so both A and B describe allowed contents and C combines them. This is not permission to begin with a digit or to use a reserved word: those separate restrictions still apply. Module support: PDF p. 5.

Why each choice is right or wrong

A - Incorrect: "alphanumeric characters" does not match the rule or result derived here. The permitted identifier characters are letters, digits and underscore (p.5), so both A and B describe allowed contents and C combines them. This is not permission to begin with a digit or to use a reserved word: those separate restrictions still apply.

B - Incorrect: "underscores" does not match the rule or result derived here. The permitted identifier characters are letters, digits and underscore (p.5), so both A and B describe allowed contents and C combines them. This is not permission to begin with a digit or to use a reserved word: those separate restrictions still apply.

C - Correct: "Both A and B" matches the derived result. The permitted identifier characters are letters, digits and underscore (p.5), so both A and B describe allowed contents and C combines them. This is not permission to begin with a digit or to use a reserved word: those separate restrictions still apply.

D - Incorrect: "None of the above" does not match the rule or result derived here. The permitted identifier characters are letters, digits and underscore (p.5), so both A and B describe allowed contents and C combines them. This is not permission to begin with a digit or to use a reserved word: those separate restrictions still apply.

Primary note proof

Study-note section CP05

> In the basic ASCII identifier rules, start with a letter or underscore; continue with letters, digits or underscores. Do not use a keyword. C++ is case sensitive: exitExam and ExitExam differ; delete and float are keywords, but FLOAT is a legal identifier. variable_1234 is valid; 4thInput, T-3 and names containing spaces are not single identifiers. Avoid reserved underscore patterns in application code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 2-10

> 1. Keywords (reserved words) - Unique meaning within a C++ program and must not be used for any other purposes. E.g., if, namespace for, switch, continue, float, new, default, break Etc. 2. Identifiers - Name associated with a function or data object and used to refer to that function or data object. An identifier must: o Start with a letter or underscore o Consist only of letters, the digits 0-9, or the underscore symbol _ o Not be a reserved word

Question sources

Web Programming practice questions · Question 63

Question 54 | Q01140

What does "cout" represent?

A. Output statement

B. Input statement

C. Compiler

D. None of the above

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: Output statement

The module names cout the standard output stream and shows cout<<object displaying information. Output statement is the closest offered description; cout itself is a stream object. cin handles input, while a compiler translates source. Module support: PDF p. 6.

Why each choice is right or wrong

A - Correct: "Output statement" matches the derived result. The module names cout the standard output stream and shows cout<<object displaying information. Output statement is the closest offered description; cout itself is a stream object. cin handles input, while a compiler translates source.

B - Incorrect: "Input statement" does not match the rule or result derived here. The module names cout the standard output stream and shows cout<<object displaying information. Output statement is the closest offered description; cout itself is a stream object. cin handles input, while a compiler translates source.

C - Incorrect: "Compiler" does not match the rule or result derived here. The module names cout the standard output stream and shows cout<<object displaying information. Output statement is the closest offered description; cout itself is a stream object. cin handles input, while a compiler translates source.

D - Incorrect: "None of the above" does not match the rule or result derived here. The module names cout the standard output stream and shows cout<<object displaying information. Output statement is the closest offered description; cout itself is a stream object. cin handles input, while a compiler translates source.

Primary note proof

Study-note section CP28

> Include <iostream> and qualify std::cin/std::cout or introduce the names explicitly. cout is the standard output stream; << inserts values. cin is standard input; >> extracts formatted values. std::endl inserts a newline and flushes. C++ names are case sensitive: Cout, Int and std: :cout are not equivalent valid spellings of cout, int and std::cout.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 6 | lines 4-15

> 6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Computer Science Exit Exam practice questions · Question 1

Question 55 | Q01141

Which terms can denote a defect in a program, rather than a debugging tool or the act of correcting defects? Select all that apply.

A. Bug

B. Debugger

C. Error

D. Debug

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A, C: A. Bug; C. Error

A bug is a defect in a program, and error can also name such a defect in this question's terminology. Both nouns therefore fit the requested meaning. A debugger instead names a tool used to investigate execution, and debug names the activity of locating or correcting defects.

Why each choice is right or wrong

A - Correct: Bug denotes a program defect.

B - Incorrect: A debugger is a tool for investigating defects.

C - Correct: Error can denote the defect itself.

D - Incorrect: Debug describes the correction/investigation activity rather than the defect.

Primary note proof

Study-note section CP32

> A syntax error violates grammar; a type error can be syntactically well formed but rejected during semantic checking. A runtime error occurs during execution; a logic error gives the wrong behavior. A bug is a defect; debugging locates and corrects it. Read the first useful compiler diagnostic, inspect nearby tokens and declarations, repair and rebuild; a debugger is useful after a runnable program exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 2-18

> 1. Introduction to programming Computer programming is the process of writing, testing, debugging/troubleshooting, and maintaining the source code of computer programs. Programs are the instructions that tells the computer what to do .so, writing computer programs means writing instructions, that will make the computer follow and run a program based on those instructions. We are going to write computer programs to solve problems. Problem Solving Problem: gap between the existing and the desired or proposed situation where problem solving will try to fill this gap. Problem solving: The process of transforming the description of a problem into the solution by using our knowledge of the problem-solving strategies, techniques and tools. Problem solving techniques • Analyze the problem: Carefully understand the problem • Outline the problem requirements • Does program require user interaction? • Does program manipulate data? • What is the output? • Design steps (algorithm) to solve the problem

Question sources

Computer Science Exit Exam practice questions · Question 2

Question 56 | Q01142

is a language translators required to translate the assembly languages into machine code.

A. | a | Compiler x

B. ? | Assembler 4

C. |@| Interpreter x

D. | None of the above x

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: Assembler.

Compilation may emit assembly, and the assembler emits machine instructions in an object file. The translator named by the question is the assembler.

Why each choice is right or wrong

A - Incorrect: The compiler can produce assembly from higher-level source; that is the preceding stage in CP04.

B - Correct: The assembler translates assembly instructions into machine instructions contained in an object file.

C - Incorrect: An interpreter executes a program representation instead of performing the stated assembly-to-object translation.

D - Incorrect: Assembler is present and directly matches the translator role, so none of the above is false.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before

execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Computer Programming module | PDF page 4 | lines 21-30

The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

Computer Science Exit Exam practice questions · Question 3

Question 57 | Q01144

An error in the statement in a computer program that violates the rules of the programming.
a | Logical error x

A. ? | Syntax error ov

B. '@| Semantic error x

C. | All of the above x

D. 6 - True or false

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: ? | Syntax error ov

Syntax is the set of rules governing use of language symbols, while semantics concerns the meanings of syntactically correct constructs (p.3). Violating the written language rules is therefore a syntax error. Module support: PDF p. 3.

Why each choice is right or wrong

A - Correct: Syntax error matches violation of the language rules defined on p.3.

B - Incorrect: Semantic error concerns meaning rather than the syntactic rule violation described (p.3).

C - Incorrect: All would include semantic/logical alternatives, which are not equivalent to violating syntax.

D - Incorrect: This is an unrelated extracted True/false heading, not an error category.

Primary note proof

Study-note section CP32

A syntax error violates grammar; a type error can be syntactically well formed but rejected during semantic checking. A runtime error occurs during execution; a logic error gives the wrong behavior. A bug is a defect; debugging locates and corrects it. Read the first useful compiler diagnostic, inspect nearby tokens and declarations, repair and rebuild; a debugger is useful after a runnable program exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Computer Science Exit Exam practice questions · Question 5

Question 58 | Q01145

Keywords are words whose meanings are defined to the compiler.

A. True

B. False

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: True

The module calls keywords reserved words with a unique meaning in a C++ program and forbids other uses (p.5). This supports True; False contradicts their predefined role. Module support: PDF p. 5.

Why each choice is right or wrong

A - Correct: "True" matches the derived result. The module calls keywords reserved words with a unique meaning in a C++ program and forbids other uses (p.5). This supports True; False contradicts their predefined role.

B - Incorrect: "False" does not match the rule or result derived here. The module calls keywords reserved words with a unique meaning in a C++ program and forbids other uses (p.5). This supports True; False contradicts their predefined role.

Primary note proof

Study-note section CP-S03

> A keyword is a reserved word with a predefined meaning in the language grammar. It cannot be repurposed as an ordinary variable or function name. An identifier is a name used to refer to a data object or function; its spelling must obey the identifier-character rules and avoid keywords.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 2-10

> 1. Keywords (reserved words) - Unique meaning within a C++ program and must not be used for any other purposes. E.g., if, namespace for, switch, continue, float, new, default, break Etc. 2. Identifiers - Name associated with a function or data object and used to refer to that function or data object. An identifier must: o Start with a letter or underscore o Consist only of letters, the digits 0-9, or the underscore symbol _ o Not be a reserved word

Question sources

Computer Science Exit Exam practice questions · Question 6

Question 59 | Q01146

In which constant that can be written in a single quotes?

A. | a | String constant x

B. i? Character constant vA

C. /@| All of the above x

D. | None of the above x

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer B: i? Character constant vA

The module examples use single quotes for a character such as A and double quotes for a string such as Hello There (pp.8,22). Thus Character constant is the correct category. String constant uses double quotes, so All and None fail. Module support: PDF p. 8, PDF p. 22.

Why each choice is right or wrong

A - Incorrect: "| a | String constant x" does not match the rule or result derived here. The module examples use single quotes for a character such as A and double quotes for a string such as Hello There (pp.8,22). Thus Character constant is the correct category. String constant uses double quotes, so All and None fail.

B - Correct: "i? Character constant vA" matches the derived result. The module examples use single quotes for a character such as A and double quotes for a string such as Hello There (pp.8,22). Thus Character constant is the correct category. String constant uses double quotes, so All and None fail.

C - Incorrect: "/@| All of the above x" does not match the rule or result derived here. The module examples use single quotes for a character such as A and double quotes for a string such as Hello There (pp.8,22). Thus Character constant is the correct category. String constant uses double quotes, so All and None fail.

D - Incorrect: "| None of the above x" does not match the rule or result derived here. The module examples use single quotes for a character such as A and double quotes for a string such as Hello There (pp.8,22). Thus Character constant is the correct category. String constant uses double quotes, so All and None fail.

Primary note proof

Study-note section CP07

> Single quotes denote a character literal, such as 'A'; double quotes denote a string literal, such as "A". A string includes more storage than just its visible characters when a terminator is required. sizeof reports size in bytes, with sizeof(char)==1; the byte width and other sizes are implementation dependent. sizeof(array) gives total array storage before conversion to a pointer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 8 | lines 27-33

> Constant: A constant is any expression that has a fixed value like integer, float, character and string Eg. 120, 15.2, „A‟, “Hello There” Generally, we have two ways to define constant: 1. Using Const Key word Syntax: const type constant name=value; e.g. const float PI=3.14; 2. Using #define Keyword Syntax: #define constant name value e.g. #define PI 3.14

Computer Programming module | PDF page 22 | lines 22-38

> String and pointer String: - is an ordered sequence of characters - Sequence of character in which the last character is the null character „\0‟ - The null character indicates the end of the string - C++ provides following two types of string representations: o The C-style character string. o The string class type introduced with Standard C++. NB: →If a string has n characters, then it requires an n+1 element array (at least) to store it. →In order to use the string data type, the C++ string header <string> must be included at the top of the program The C-Style Character String - originated within the C language and continues to be supported within C++ - array of characters which is terminated by a null character '\0' Example: char status[5]={„G‟,‟O‟,‟O‟,‟D‟,‟\0‟};//char status[]=”Good”; cout<<status;

Question sources

Computer Science Exit Exam practice questions · Question 7

Question 60 | Q01147

In which constant that can be written in a double quotes?

A. a | String constants oS

B. i? Character constants x

C. |@| All of the above x

D. | None of the above x

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer A: a | String constants oS

The examples enclose strings in double quotes and characters in single quotes (pp.8,22). Therefore String constants is the intended category; Character, All and None do not fit the shown notation. Module support: PDF p. 8, PDF p. 22.

Why each choice is right or wrong

A - Correct: "a | String constants oS" matches the derived result. The examples enclose strings in double quotes and characters in single quotes (pp.8,22). Therefore String constants is the intended category; Character, All and None do not fit the shown notation.

B - Incorrect: "i? Character constants x" does not match the rule or result derived here. The examples enclose strings in double quotes and characters in single quotes (pp.8,22). Therefore String constants is the intended category; Character, All and None do not fit the shown notation.

C - Incorrect: "|@| All of the above x" does not match the rule or result derived here. The examples enclose strings in double quotes and characters in single quotes (pp.8,22). Therefore String constants is the intended category; Character, All and None do not fit the shown notation.

D - Incorrect: "| None of the above x" does not match the rule or result derived here. The examples enclose strings in double quotes and characters in single quotes (pp.8,22). Therefore String constants is the intended category; Character, All and None do not fit the shown notation.

Primary note proof

Study-note section CP07

Single quotes denote a character literal, such as 'A'; double quotes denote a string literal, such as "A". A string includes more storage than just its visible characters when a terminator is required. sizeof reports size in bytes, with sizeof(char)==1; the byte width and other sizes are implementation dependent. sizeof(array) gives total array storage before conversion to a pointer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 8 | lines 27-33

Constant: A constant is any expression that has a fixed value like integer, float, character and string Eg. 120, 15.2, „A‟, “Hello There” Generally, we have two ways to define constant: 1. Using Const Key word Syntax: const type constant name=value; e.g. const float PI=3.14; 2. Using #define Keyword Syntax: #define constant name value e.g. #define PI 3.14

Computer Programming module | PDF page 22 | lines 22-38

String and pointer String: - is an ordered sequence of characters - Sequence of character in which the last character is the null character „\0‟ - The null character indicates the end of the string - C++ provides following two types of string representations: o The C-style character string. o The string class type introduced with Standard C++. NB: →If a string has n characters, then it requires an n+1 element array (at least) to store it. →In order to use the string data type, the C++ string header <string> must be included at the top of the program The C-Style Character String - originated within the C language and continues to be supported within C++ - array of characters which is terminated by a null character '\0' Example: char status[5]={„G‟,‟O‟,‟O‟,‟D‟,‟\0‟};//char status[]=”Good”; cout<<status;

Question sources

Computer Science Exit Exam practice questions · Question 8

Question 61 | Q01148

What is the output of the following C++ fragment?

```
#include <iostream>
using namespace std;
int main()
{
int a = 8;
int b = 5;
int c;
c = a++ + --b;
cout << c << endl;
return 0;
}
```

A. 13

B. 15

C. 11

D. 12

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer D: 12

Postfix a++ contributes 8 to the expression and then makes a=9. Prefix --b changes 5 to 4 before use. Thus c=8+4=12. Thirteen uses old b; the other sums do not follow the two operator timings. Module support: PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "13" does not match the rule or result derived here. Postfix a++ contributes 8 to the expression and then makes a=9. Prefix --b changes 5 to 4 before use. Thus c=8+4=12. Thirteen uses old b; the other sums do not follow the two operator timings.

B - Incorrect: "15" does not match the rule or result derived here. Postfix a++ contributes 8 to the expression and then makes a=9. Prefix --b changes 5 to 4 before use. Thus c=8+4=12. Thirteen uses old b; the other sums do not follow the two operator timings.

C - Incorrect: "11" does not match the rule or result derived here. Postfix a++ contributes 8 to the expression and then makes a=9. Prefix --b changes 5 to 4 before use. Thus c=8+4=12. Thirteen uses old b; the other sums do not follow the two operator timings.

D - Correct: "12" matches the derived result. Postfix a++ contributes 8 to the expression and then makes a=9. Prefix --b changes 5 to 4 before use. Thus c=8+4=12. Thirteen uses old b; the other sums do not follow the two operator timings.

Primary note proof

Study-note section CP09

Prefix ++x changes x and yields the new value; postfix x++ yields the previous value and also increments x. Separate the value used in the expression from the stored value after it. Starting a=8,b=5, a++ + --b uses 8+4, gives 12, and leaves a=9,b=4. Starting x=1,y=10,z=-3, (x++*y--)+ ++z gives 8 and leaves x=2,y=9,z=-2.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, | |) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 | | 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science Exit Exam practice questions · Question 9

Question 62 | Q01149

Which data type represents logical values?

A. Integer

B. Boolean

C. Double

D. Float

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: Boolean.

CP06 assigns bool to logical values, whereas int represents whole numbers and float/double represent fractional approximations. Boolean is therefore the intended type category. Numeric values can participate in truth conversion, but that does not make their numeric types the dedicated representation of logical true/false values.

Why each choice is right or wrong

A - Incorrect: Integer represents whole numbers; possible conversion to truth does not make it the logical type.

B - Correct: Boolean represents logical true/false values, matching bool in the note's C++ type mapping.

C - Incorrect: Double represents fractional numeric approximations rather than the dedicated logical type.

D - Incorrect: Float represents fractional numeric approximations rather than the dedicated logical type.

Primary note proof

Study-note section CP12

> A standalone if may execute independently of the following if. An else binds to its nearest unmatched if; an else-if ladder executes the first true branch only. In nested if(c1){if(c2) S1; else S2;}, S2 requires c1 true and c2 false. Braces determine grouping.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 5 | lines 24-33

> 5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Question sources

Computer Science Exit Exam practice questions · Question 10

Question 63 | Q01150

What is the output of the following C++ fragment?

```
#include <iostream>
using namespace std;
int main()
{
int x = 25;
if (x < 20)
cout << "Low Quality" << endl;
else
cout << "High Quality" << endl;
return 0;
}
```

A. Low Quality

B. High Quality

C. No output

D. None of the above

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer B: High Quality

The condition 25<20 is false, so the else branch prints High Quality. The then message is skipped; a defined else branch means the no-output alternative is false, and B supplies a matching option. Module support: PDF p. 9.

Why each choice is right or wrong

A - Incorrect: "Low Quality" does not match the rule or result derived here. The condition 25<20 is false, so the else branch prints High Quality. The then message is skipped; a defined else branch means the no-output alternative is false, and B supplies a matching option.

B - Correct: "High Quality" matches the derived result. The condition 25<20 is false, so the else branch prints High Quality. The then message is skipped; a defined else branch means the no-output alternative is false, and B supplies a matching option.

C - Incorrect: "No output" does not match the rule or result derived here. The condition 25<20 is false, so the else branch prints High Quality. The then message is skipped; a defined else branch means the no-output alternative is false, and B supplies a matching option.

D - Incorrect: "None of the above" does not match the rule or result derived here. The condition 25<20 is false, so the else branch prints High Quality. The then message is skipped; a defined else branch means the no-output alternative is false, and B supplies a matching option.

Primary note proof

Study-note section CP06

Use int for whole numbers, float/double for fractional approximations, char for a character and bool for logical values in C++. Type sizes depend on the implementation; do not import Java widths into C++. short n, short int n and int short n denote the same signed short type. const int n=5 and int const n=5 are equivalent; an ordinary const int needs initialization. typedef creates an alias, not a distinct new type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 9 | lines 9-21

Conditional Statements A. The if Statement Syntax: if (expression) statement; First expression is evaluated. If the outcome is nonzero (true) then statement is executed. Otherwise, nothing happens. if-else statement Syntax: if (expression) statement1; else statement2; First expression is evaluated. If the outcome is nonzero (true) then statement1 is executed. Otherwise, statement2 is executed.

Question sources

Computer Science Exit Exam practice questions · Question 11

Question 64 | Q01166

Loading executable code, which is the output of a linker, into main memory is called:

A. Linker

B. Assembler

C. Compiler

D. Loader

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: Loader

The module says the loader copies the linked executable into computer memory and sends execution to the CPU (p.4). Linking builds that executable; compilation translates source, so they do not name the memory-loading role. Module support: PDF p. 4.

Why each choice is right or wrong

A - Incorrect: "Linker" does not match the rule or result derived here. The module says the loader copies the linked executable into computer memory and sends execution to the CPU (p.4). Linking builds that executable; compilation translates source, so they do not name the memory-loading role.

B - Incorrect: "Assembler" does not match the rule or result derived here. The module says the loader copies the linked executable into computer memory and sends execution to the CPU (p.4).

Linking builds that executable; compilation translates source, so they do not name the memory-loading role.

C - Incorrect: "Compiler" does not match the rule or result derived here. The module says the loader copies the linked executable into computer memory and sends execution to the CPU (p.4). Linking builds that executable; compilation translates source, so they do not name the memory-loading role.

D - Correct: "Loader" matches the derived result. The module says the loader copies the linked executable into computer memory and sends execution to the CPU (p.4). Linking builds that executable; compilation translates source, so they do not name the memory-loading role.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

Computer Science Exit Exam practice questions · Question 27

Question 65 | Q01174

When control reaches a loop normally, which loop form executes its body before its first condition test, guaranteeing at least one body execution?

A. while loop

B. for loop

C. nested loop

D. do-while loop

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: D. do-while loop

A do-while loop performs its body first and tests the continuation condition afterward. Even when that first test is false, one body execution has already occurred. A while loop and a conventional for loop can reject entry on their first condition test. Nesting describes one loop inside another and supplies no independent guarantee that either body runs. The option names the post-test form.

Why each choice is right or wrong

A - Incorrect: while checks its condition before entering the body.

B - Incorrect: for can fail its initial condition and execute zero times.

C - Incorrect: A nested loop can have an outer condition that is false initially.

D - Correct: do-while performs its body before the first condition test.

Primary note proof

Study-note section CP14

> while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 2-7

> The do-while loop The condition in the do-while is evaluated after the execution of statement instead of before, granting at least one execution of statement even if condition is never fulfilled. do {statement; }while (condition); First statement is executed and then condition is evaluated. If the outcome of the latter is nonzero then the whole process is repeated. Otherwise, the loop is terminated.

Computer Programming module | PDF page 11 | lines 5-13

> Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Question sources

Computer Science Exit Exam practice questions · Question 35

Question 66 | Q01196

The _____ statement causes a loop to terminate immediately.

A. Continue

B. Goto

C. Break

D. Jump

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Break

break immediately leaves the loop for the statement outside it (p.13). continue skips the remainder of the current iteration but keeps looping. Goto names a separate label transfer, and Jump is not the module statement that directly terminates a loop. Module support: PDF p. 13.

Why each choice is right or wrong

A - Incorrect: "Continue" does not match the rule or result derived here. break immediately leaves the loop for the statement outside it (p.13). continue skips the remainder of the current iteration but keeps looping. Goto names a separate label transfer, and Jump is not the module statement that directly terminates a loop.

B - Incorrect: "Goto" does not match the rule or result derived here. break immediately leaves the loop for the statement outside it (p.13). continue skips the remainder of the current iteration but keeps looping. Goto names a separate label transfer, and Jump is not the module statement that directly terminates a loop.

C - Correct: "Break" matches the derived result. break immediately leaves the loop for the statement outside it (p.13). continue skips the remainder of the current iteration but keeps looping. Goto names a separate label transfer, and Jump is not the module statement that directly terminates a loop.

D - Incorrect: "Jump" does not match the rule or result derived here. break immediately leaves the loop for the statement outside it (p.13). continue skips the remainder of the current iteration but keeps looping. Goto names a separate label transfer, and Jump is not the module statement that directly terminates a loop.

Primary note proof

Study-note section CP15

> break exits the nearest enclosing loop or switch. continue skips the rest of the current loop iteration: a for loop then runs its update; a while loop goes directly to its condition. return leaves the current function and may supply a result. In main it ends the program normally.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 13 | lines 30-35

> Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Question sources

Computer Science model exam 2 · Question 17

Question 67 | Q01214

A translator that translates a single-line instruction at a time is:

A. Compiler

B. Assembler

C. Interpreter

D. Linker

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Interpreter

The module explicitly contrasts whole-program compilation with an interpreter translating one line at a time across pp.3-4. Therefore Interpreter fits; linking combines object files, and the other named translators are not the one-line interpreter role. Module support: PDF p. 3.

Why each choice is right or wrong

A - Incorrect: "Compiler" does not match the rule or result derived here. The module explicitly contrasts whole-program compilation with an interpreter translating one line at a time across pp.3-4. Therefore Interpreter fits; linking combines object files, and the other named translators are not the one-line interpreter role.

B - Incorrect: "Assembler" does not match the rule or result derived here. The module explicitly contrasts whole-program compilation with an interpreter translating one line at a time across pp.3-4. Therefore Interpreter fits; linking combines object files, and the other named translators are not the one-line interpreter role.

C - Correct: "Interpreter" matches the derived result. The module explicitly contrasts whole-program compilation with an interpreter translating one line at a time across pp.3-4. Therefore Interpreter fits; linking combines object files, and the other named translators are not the one-line interpreter role.

D - Incorrect: "Linker" does not match the rule or result derived here. The module explicitly contrasts whole-program compilation with an interpreter translating one line at a time across pp.3-4. Therefore Interpreter fits; linking combines object files, and the other named translators are not the one-line interpreter role.

Primary note proof

Study-note section CP04

> GCC documents the first four stages explicitly. A compiler need not emit a complete executable or even native machine code. Java javac produces bytecode, which a JVM interprets or compiles at runtime. Interpreters execute a program representation; “one source line at a time” is a beginner approximation. Compilation uses multiple passes, so “whole program in one step” is not literally universal.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Computer Science model exam 2 · Question 35

Question 68 | Q01231

Consider the following outline of a nested if-else structure:

```
if (condition1)
{
if (condition2)
statement1;
else
statement2;
}
```

Which of the statements below is true regarding this structure?

A. Syntactically, it is invalid to have more if clauses than else clauses.

B. Statement2 will only execute if condition1 is false and condition2 is false.

C. Statement2 will only execute if condition1 is true and condition2 is false.

D. Statement2 will only execute if condition1 is false, it does not matter what condition2 is.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Statement2 will only execute if condition1 is true and condition2 is false.

The outer body is entered only when condition1 is true. Inside that body, condition2 false selects its else and statement2. If condition1 is false neither nested branch executes. An if may omit else, so A imposes an unsupported syntactic requirement. Module support: PDF p. 9.

Why each choice is right or wrong

A - Incorrect: "Syntactically, it is invalid to have more if clauses than else clauses." does not match the rule or result derived here. The outer body is entered only when condition1 is true. Inside that body, condition2 false selects its else and statement2. If condition1 is false neither nested branch executes. An if may omit else, so A imposes an unsupported syntactic requirement.

B - Incorrect: "Statement2 will only execute if condition1 is false and condition2 is false." does not match the rule or result derived here. The outer body is entered only when condition1 is true. Inside that body, condition2 false selects its else and statement2. If condition1 is false neither nested branch executes. An if may omit else, so A imposes an unsupported syntactic requirement.

C - Correct: "Statement2 will only execute if condition1 is true and condition2 is false." matches the derived result. The outer body is entered only when condition1 is true. Inside that body, condition2 false selects its else and statement2. If condition1 is false neither nested branch executes. An if may omit else, so A imposes an unsupported syntactic requirement.

D - Incorrect: "Statement2 will only execute if condition1 is false, it does not matter what condition2 is." does not match the rule or result derived here. The outer body is entered only when condition1 is true. Inside that body, condition2 false selects its else and statement2. If condition1 is false neither nested branch executes. An if may omit else, so A imposes an unsupported syntactic requirement.

Primary note proof

Study-note section CP12

> A standalone if may execute independently of the following if. An else binds to its nearest unmatched if; an else-if ladder executes the first true branch only. In nested if(c1){if(c2) S1; else S2;}, S2 requires c1 true and c2 false. Braces determine grouping.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 9 | lines 9-21

> Conditional Statements A. The if Statement Syntax: if (expression) statement; First expression is evaluated. If the outcome is nonzero (true) then statement is executed. Otherwise, nothing happens. if-else statement Syntax: if (expression) statement1; else statement2; First expression is evaluated. If the outcome is nonzero (true) then statement1 is executed. Otherwise, statement2 is executed.

Question sources

Computer Science model exam 2 · Question 52

Question 69 | Q01256

Which of the following is a valid variable name?

A. T-3

B. GGGGG

C. float

D. 4thInput

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: GGGGG

GGGGG starts with a letter, uses only letters, and is not one of the given reserved keywords. T-3 contains a hyphen, float is explicitly reserved, and 4thInput starts with a digit; all violate p.5 rules. Module support: PDF p. 5.

Why each choice is right or wrong

A - Incorrect: "T-3" does not match the rule or result derived here. GGGGG starts with a letter, uses only letters, and is not one of the given reserved keywords. T-3 contains a hyphen, float is explicitly reserved, and 4thInput starts with a digit; all violate p.5 rules.

B - Correct: "GGGGG" matches the derived result. GGGGG starts with a letter, uses only letters, and is not one of the given reserved keywords. T-3 contains a hyphen, float is explicitly reserved, and 4thInput starts with a digit; all violate p.5 rules.

C - Incorrect: "float" does not match the rule or result derived here. GGGGG starts with a letter, uses only letters, and is not one of the given reserved keywords. T-3 contains a hyphen, float is explicitly reserved, and 4thInput starts with a digit; all violate p.5 rules.

D - Incorrect: "4thInput" does not match the rule or result derived here. GGGGG starts with a letter, uses only letters, and is not one of the given reserved keywords. T-3 contains a hyphen, float is explicitly reserved, and 4thInput starts with a digit; all violate p.5 rules.

Primary note proof

Study-note section CP05

In the basic ASCII identifier rules, start with a letter or underscore; continue with letters, digits or underscores. Do not use a keyword. C++ is case sensitive: exitExam and ExitExam differ; delete and float are keywords, but FLOAT is a legal identifier. variable_1234 is valid; 4thInput, T-3 and names containing spaces are not single identifiers. Avoid reserved underscore patterns in application code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 2-10

> 1. Keywords (reserved words) - Unique meaning within a C++ program and must not be used for any other purposes. E.g., if, namespace for, switch, continue, float, new, default, break Etc. 2. Identifiers - Name associated with a function or data object and used to refer to that function or data object. An identifier must: o Start with a letter or underscore o Consist only of letters, the digits 0-9, or the underscore symbol _ o Not be a reserved word

Question sources

Computer Science model exam 2 · Question 77

Question 70 | Q01259

An array is declared and initialized as follows:

int A[4][4] = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 8, 7, 6}, {5, 4, 3, 2}};

What are the values of `A[1][1]`, `A[2][3]`, and `A[3][3]`, respectively?

A. 7, 6, 3

B. 6, 6, 2

C. 6, 2, 6

D. 2, 6, 6

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: 6, 6, 2

Arrays are zero-indexed (p.19). Row 1 is {5,6,7,8}, hence A[1][1]=6; row 2 is {9,8,7,6}, hence A[2][3]=6; row 3 ends with 2, hence A[3][3]=2. Only B preserves all three positions. Module support: PDF p. 19, PDF p. 21.

Why each choice is right or wrong

A - Incorrect: "7, 6, 3" does not match the rule or result derived here. Arrays are zero-indexed (p.19). Row 1 is {5,6,7,8}, hence A[1][1]=6; row 2 is {9,8,7,6}, hence A[2][3]=6; row 3 ends with 2, hence A[3][3]=2. Only B preserves all three positions.

B - Correct: "6, 6, 2" matches the derived result. Arrays are zero-indexed (p.19). Row 1 is {5,6,7,8}, hence A[1][1]=6; row 2 is {9,8,7,6}, hence A[2][3]=6; row 3 ends with 2, hence A[3][3]=2. Only B preserves all three positions.

C - Incorrect: "6, 2, 6" does not match the rule or result derived here. Arrays are zero-indexed (p.19). Row 1 is {5,6,7,8}, hence A[1][1]=6; row 2 is {9,8,7,6}, hence A[2][3]=6; row 3 ends with 2, hence A[3][3]=2. Only B preserves all three positions.

D - Incorrect: "2, 6, 6" does not match the rule or result derived here. Arrays are zero-indexed (p.19). Row 1 is {5,6,7,8}, hence A[1][1]=6; row 2 is {9,8,7,6}, hence A[2][3]=6; row 3 ends with 2, hence A[3][3]=2. Only B preserves all three positions.

Primary note proof

Study-note section CP23

int a[R][C] has R rows, each with C elements. a[r][c] uses zero-based row then column. Its element offset is r*C+c in row-major storage. In the 4x4 matrix example, a[1][1]=6, a[2][3]=6 and a[3][3]=2. In Java, length belongs to each array; a traversal ending at arr.length-2 visits fewer elements than a full traversal.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 19 | lines 4-29

Arrays - an array is a collection of fixed-size number of elements of the same type stored sequentially in memory. - An individual element of an array is identified by its own unique index (or subscript). - The number of elements must be an integer. - The size of the array is referred to as its dimension - Array in C++ is Zero indexed. Declaring of one-dimensional Array Syntax: elementstype arrayname [size]; → You can use constants such as macro definitions or const ints as the sizes of arrays. Example: int num[5]; string fruit[3]; #define numstudents 100 or const int numstudents=100; double Score[numstudents] ; double Score[numstudents] ; Initialization of an Array - Arrays can be initialized at declaration time in two different ways: Syntax: elementstype arrayname[size] = {list each value separated by comma}; E.g int num[5]={1,2,3,4,5}; elementstype arrayname[ ] = { list with any number of values }; E.g int num[]={1,2,3,4,5}; string fruit[]={“Apple”,“orange”, “Mango”}; Accessing Array Elements - Each element could be accessed using an index into the array - When accessing an array, the index given must be a positive integer from 0 to n-1, where n is the dimension of the array.

Computer Programming module | PDF page 21 | lines 28-36

Multidimensional Array C++ support the creation of multidimensional array, through the addition of more than one set of brackets. Syntax: elementstype arrayname [size1] [size2] ... [sizen]; Two-Dimensional Array Declaration: elementstype arrayname [size1] [size2]; Eg. int num[2][3];//declare 2 row, 3 columns to store 6 elements Initialization of Two-Dimensional Array E.g. int num[2][3]={1,2,3,4,5,6};//2 row 3 columns Or int num[2][3]={{1,2},{3,4},{5,6}};

Question sources

Computer Science model exam 2 · Question 80

Question 71 | Q01275

Before a program is written, the programmer must ensure to:

A. Clear and understand what data are to be used

B. Clear and understand the desired result

C. Clear and understand the procedure to be used

D. All of the above

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: All of the above

The problem-solving checklist requires understanding the problem, its data and output, and designing algorithm steps (p.2). Thus data, desired result and procedure all belong to preparation, making All the most complete choice. A/B/C are individually necessary but incomplete as a combined checklist. Module support: PDF p. 2.

Why each choice is right or wrong

A - Incorrect: Supported component: understand the data to be manipulated (p.2); less complete than All.

B - Incorrect: Supported component: identify expected output (p.2); less complete than All.

C - Incorrect: Supported component: design the algorithm (p.2); less complete than All.

D - Correct: Combines all three preparation requirements on p.2.

Primary note proof

Study-note section CP01

> Identify inputs, outputs, legal ranges and edge cases before coding. Solve one small case manually, choose an algorithm, express it as pseudocode or a flowchart, implement, test and maintain. For the largest of three numbers, initialize best to the first input, compare the second and third, then report best. Initializing best to zero fails when all inputs are negative.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 2-18

> 1. Introduction to programming Computer programming is the process of writing, testing, debugging/troubleshooting, and maintaining the source code of computer programs. Programs are the instructions that tells the computer what to do .so, writing computer programs means writing instructions, that will make the computer follow and run a program based on those instructions. We are going to write computer programs to solve problems. Problem Solving Problem: gap between the existing and the desired or proposed situation where problem solving will try to fill this gap. Problem solving: The process of transforming the description of a problem

into the solution by using our knowledge of the problem-solving strategies, techniques and tools. Problem solving techniques • Analyze the problem: Carefully understand the problem • Outline the problem requirements • Does program require user interaction? • Does program manipulate data? • What is the output? • Design steps (algorithm) to solve the problem

Question sources

Computer Science model exam 2 · Question 96

Question 72 | Q01393

In the conventional language-level taxonomy, which category generally offers source that is easier to read and often portable after recompilation when platform assumptions are avoided?

A. Low-level Language

B. High-Level Language

C. Decimal-Format

D. Middle-Level Language

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: B. High-Level Language

High-level source is easier to read and often portable after recompilation when it avoids platform assumptions. Machine and assembly languages depend more directly on an instruction set. The question asks about this language-level property, rather than claiming that people write programs in only one category. Decimal format describes numeric notation, and middle-level is not a separate category in the specified taxonomy.

Why each choice is right or wrong

A - Incorrect: Low-level machine and assembly source depends on an instruction set.

B - Correct: High-level source has the readability and conditional portability described.

C - Incorrect: Decimal format is a numeric representation, not this language category.

D - Incorrect: The stated taxonomy assigns this property to high-level languages.

Primary note proof

Study-note section CP03

Machine and assembly code depend on an instruction set. High-level source is easier to read and often portable after recompilation, provided it avoids platform assumptions. People can write both assembly and high-level programs. Python, Perl and PHP are commonly called scripting languages; C++ is conventionally compiled. These labels describe common use, not an absolute technical prohibition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Compiler Design review questions · Question 1

Question 73 | Q01394

Which computer program accepts the high-level language and converts it into assembly language?

A. Interpreter

B. Linker

C. Assembler

D. Compiler

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Compiler.

CP04 describes compilation as a stage that may emit assembly from source. The assembler then translates that assembly into object-file machine instructions, and linking combines objects and libraries. Thus the question asks for the compiler, without implying that every compiler implementation must produce assembly as its output.

Why each choice is right or wrong

A - Incorrect: An interpreter executes a representation rather than being the stated high-level-to-assembly translation stage.

B - Incorrect: The linker resolves external symbols and combines objects and libraries after their translation.

C - Incorrect: The assembler consumes assembly and emits machine instructions, rather than consuming high-level source.

D - Correct: A compiler can translate high-level source into assembly, as described in the conventional toolchain.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

Compiler Design review questions · Question 2

Question 74 | Q01395

In the conventional ahead-of-time C++ toolchain, is the source translation unit compiled before its resulting program is executed, even though compilation may use several internal passes?

A. No

B. Depends on the Compiler

C. Don't Know

D. Yes

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: D. Yes

The conventional C++ toolchain preprocesses and compiles source, assembles object files, links an executable and then loads it for execution. Thus the specified source translation occurs before running the resulting program. This does not mean the compiler has only one internal pass or must directly emit a complete native executable. Under the explicit conventional ahead-of-time model, Yes is the definite answer.

Why each choice is right or wrong

A - Incorrect: No contradicts the stipulated ahead-of-time build sequence.

B - Incorrect: The question fixes the conventional ahead-of-time model rather than leaving the execution strategy unspecified.

C - Incorrect: The stated model supplies enough information to decide.

D - Correct: The source unit is compiled before the resulting program executes.

Primary note proof

Study-note section CP04

> GCC documents the first four stages explicitly. A compiler need not emit a complete executable or even native machine code. Java javac produces bytecode, which a JVM interprets or compiles at runtime. Interpreters execute a program representation; “one source line at a time” is a beginner approximation. Compilation uses multiple passes, so “whole program in one step” is not literally universal.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

Compiler Design review questions · Question 3

Question 75 | Q01396

Which of the following file is an output of the assembler?

A. Program file

B. Object file

C. Data File

D. Task File

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: Object file.

In CP04's toolchain the assembler emits machine instructions in an object file. Linking subsequently combines object files and libraries into an executable. The question therefore asks for the object file, rather than an unspecified program, data or task file; those other names do not identify the assembler's defined output stage.

Why each choice is right or wrong

A - Incorrect: Program file is a vague description and does not identify the assembler's specified object-file output.

B - Correct: The assembler emits an object file containing translated machine instructions for later linking.

C - Incorrect: Data file does not identify the translated instruction object produced at the assembly stage.

D - Incorrect: Task file is not a stage output identified for the assembler in CP04.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other

object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

Compiler Design review questions · Question 4

Question 76 | Q01398

What is the linker's role in the conventional C++ build process?

A. It translates assembly source into object files.

B. It combines object files and libraries and resolves external symbols to create an executable module.

C. It is the same as the loader

D. None of the above

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: B. It combines object files and libraries and resolves external symbols to create an executable module.

The assembler has already produced object files when linking begins. The linker combines those files and libraries and resolves their external symbol references to create the executable module. The loader subsequently places that executable and required components in memory. These distinct responsibilities make option B precise: A describes assembly instead, C conflates linking with loading, and D is false because B supplies the correct role.

Why each choice is right or wrong

A - Incorrect: Translating assembly into an object file is the assembler's role.

B - Correct: The linker combines objects/libraries and resolves external references to make the executable.

C - Incorrect: The loader's memory-placement role is separate from linking.

D - Incorrect: B gives the correct linker responsibility.

Primary note proof

Study-note section CP04

In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

Compiler Design review questions · Question 6

Question 77 | Q01424

Which statement is false about the conventional C++ toolchain described in the note?

A. Every compiler directly emits a complete executable without a separate linking stage.

B. A debugger is useful after a runnable program exists.

C. Source syntax errors must be repaired for successful translation.

D. Compilation may use multiple passes and may emit assembly.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: A. Every compiler directly emits a complete executable without a separate linking stage.

The note separates several translation and build stages. Compilation can emit assembly, assembly produces object files, and linking resolves external symbols and combines those objects. It therefore does not follow that every compiler directly produces a complete executable. Source grammar errors must be corrected for successful translation, and a debugger is useful after a runnable program exists. These stage-based claims avoid unspecified execution-time and debugging-speed comparisons.

Why each choice is right or wrong

A - Correct: A compiler need not emit a complete executable; linking is a separate stage.

B - Incorrect: The note places debugger use after a runnable program exists.

C - Incorrect: A successfully translated C++ program has passed its source syntax checks.

D - Incorrect: Compilation can use multiple passes and can emit assembly.

Primary note proof

Study-note section CP04

In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Study-note section CP32

A syntax error violates grammar; a type error can be syntactically well formed but rejected during semantic checking. A runtime error occurs during execution; a logic error gives the wrong behavior. A bug is a defect; debugging locates and corrects it. Read the first useful compiler diagnostic, inspect nearby tokens and declarations, repair and rebuild; a debugger is useful after a runnable program exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Computer Programming module | PDF page 3 | lines 8-17

Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Compiler Design review questions · Question 32

Question 78 | Q01518

Which one of the following is a VALID C++ declaration statement?

A. age int;

B. int 1x, 2x;

C. int FLOAT;

D. float abebe, A-Z;

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer C: int FLOAT;

The required declaration form is type followed by identifiers (p.5). int FLOAT fits: uppercase FLOAT is used as an identifier, distinct from lowercase reserved float in the intended C++ reading. A reverses type/name, B starts identifiers with digits, and D uses a hyphen in A-Z. Module support: PDF p. 5, PDF p. 5.

Why each choice is right or wrong

A - Incorrect: "age int;" does not match the rule or result derived here. The required declaration form is type followed by identifiers (p.5). int FLOAT fits: uppercase FLOAT is used as an identifier, distinct from lowercase reserved float in the intended C++ reading. A reverses type/name, B starts identifiers with digits, and D uses a hyphen in A-Z.

B - Incorrect: "int 1x, 2x;" does not match the rule or result derived here. The required declaration form is type followed by identifiers (p.5). int FLOAT fits: uppercase FLOAT is used as an identifier, distinct from lowercase reserved float in the intended C++ reading. A reverses type/name, B starts identifiers with digits, and D uses a hyphen in A-Z.

C - Correct: "int FLOAT;" matches the derived result. The required declaration form is type followed by identifiers (p.5). int FLOAT fits: uppercase FLOAT is used as an identifier, distinct from lowercase reserved float in the intended C++ reading. A reverses type/name, B starts identifiers with digits, and D uses a hyphen in A-Z.

D - Incorrect: "float abebe, A-Z;" does not match the rule or result derived here. The required declaration form is type followed by identifiers (p.5). int FLOAT fits: uppercase FLOAT is used as an identifier, distinct from lowercase reserved float in the intended C++ reading. A reverses type/name, B starts identifiers with digits, and D uses a hyphen in A-Z.

Primary note proof

Study-note section CP05

> In the basic ASCII identifier rules, start with a letter or underscore; continue with letters, digits or underscores. Do not use a keyword. C++ is case sensitive: exitExam and ExitExam differ; delete and float are keywords, but FLOAT is a legal identifier. variable_1234 is valid; 4thInput, T-3 and names containing spaces are not single identifiers. Avoid reserved underscore patterns in application code.

Study-note section CP06

Write type name; or type name = value; for ordinary variables: int age = 20;. Declaration introduces a name and type; initialization gives its initial value; later assignment replaces a value. a=b copies a scalar value, so later changes to b do not change a. A pointer or Java object reference instead copies an address/reference.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 24-33

5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Computer Programming module | PDF page 5 | lines 2-10

1. Keywords (reserved words) - Unique meaning within a C++ program and must not be used for any other purposes. E.g., if, namespace for, switch, continue, float, new, default, break Etc. 2. Identifiers - Name associated with a function or data object and used to refer to that function or data object. An identifier must: o Start with a letter or underscore o Consist only of letters, the digits 0-9, or the underscore symbol _ o Not be a reserved word

Question sources

Mock exam 1 · Question 34

Question 79 | Q01519

Which of the following is not true?

A. (8.0 * 3) % 5 is evaluated to 4

B. 8 * 3 / 3 evaluated to 8

C. (5 > 3) || (5 != 5) && (2 <= 2) results in true

D. (7.0 * 3) / 5 is evaluated to 4.2

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: The floating-point remainder expression is invalid in C++.

The expression (8.0 * 3) has floating type, so applying % to it is invalid under CP08; it does not evaluate to 4. The integer multiplication/division in B gives 8. C is true because its left disjunct is true. D uses floating division and gives 21.0/5 = 4.2.

Why each choice is right or wrong

A - Correct: This is the false statement: floating-point operands cannot be used with the C++ remainder operator.

B - Incorrect: Equal-precedence multiplication and division group left to right: 24/3 gives the stated 8.

C - Incorrect: The first comparison is true, so the disjunction is true regardless of its right operand.

D - Incorrect: 7.0 makes the computation floating point, and 21.0 divided by 5 gives 4.2.

Primary note proof

Study-note section CP08

> Multiplication, division and remainder have higher precedence than addition and subtraction; equal-precedence arithmetic groups left to right. Integer division truncates toward zero. For integers, a == (a/b)*b + a%b when the division is valid. % requires integral operands in C++; (8.0*3)%5 is invalid. Division by zero and signed integer overflow cannot be given ordinary numeric outputs.

Study-note section CP10

> Relational and equality operators test values; && and || combine truth values; ! negates. In C++, zero converts to false and a nonzero number to true. Streams display bool as 0 or 1 unless boolalpha is set. With a=3,b=5, cout<<(a==b) prints 0.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Mock exam 1 · Question 35

Question 80 | Q01520

Which one of the following is the result of the below given code fragment?
int a = 7, b;
b = a++;
cout<<++b;

A. 6

B. 9

C. 7

D. 8

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: 8

b=a++ assigns the old a value 7 to b and increments a to 8. Then ++b changes b to 8 before output. Thus 8 prints; 7 omits the prefix increment and 6/9 do not follow the trace. Module support: PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "6" does not match the rule or result derived here. b=a++ assigns the old a value 7 to b and increments a to 8. Then ++b changes b to 8 before output. Thus 8 prints; 7 omits the prefix increment and 6/9 do not follow the trace.

B - Incorrect: "9" does not match the rule or result derived here. b=a++ assigns the old a value 7 to b and increments a to 8. Then ++b changes b to 8 before output. Thus 8 prints; 7 omits the prefix increment and 6/9 do not follow the trace.

C - Incorrect: "7" does not match the rule or result derived here. b=a++ assigns the old a value 7 to b and increments a to 8. Then ++b changes b to 8 before output. Thus 8 prints; 7 omits the prefix increment and 6/9 do not follow the trace.

D - Correct: "8" matches the derived result. b=a++ assigns the old a value 7 to b and increments a to 8. Then ++b changes b to 8 before output. Thus 8 prints; 7 omits the prefix increment and 6/9 do not follow the trace.

Primary note proof

Study-note section CP09

> Prefix ++x changes x and yields the new value; postfix x++ yields the previous value and also increments x. Separate the value used in the expression from the stored value after it. Starting a=8,b=5, a++ + --b uses 8+4, gives 12, and leaves a=9,b=4. Starting x=1,y=10,z=-3, (x++*y--)+ ++z gives 8 and leaves x=2,y=9,z=-2.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, | |) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 | | 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Mock exam 1 · Question 36

Question 81 | Q01575

Considering the below given string declaration, which one of the following statements is correct? char msg[] = "This is the final exam for C++";

A. There is an error with the initialization statement

B. strlen(msg) would return 24

C. strcpy(msg, "Hurray") would assign the content of msg to Hurray

D. cout<<msg ; would display: This is the final exam for C++

E. None of the answers are correct

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer D: cout<<msg ; would display: This is the final exam for C++

The module supports char-array string initialization and output. Printing msg displays its literal This is the final exam for C++. Its length is 28 characters, not 24. strcpy(destination,source) copies Hurray into msg, not the reverse. Thus A/B/C/E fail and D matches. Module support: PDF p. 22, PDF p. 23, PDF p. 6.

Why each choice is right or wrong

A - Incorrect: "There is an error with the initialization statement" does not match the rule or result derived here. The module supports char-array string initialization and output. Printing msg displays its literal This is the final exam for C++. Its length is 28 characters, not 24. strcpy(destination,source) copies Hurray into msg, not the reverse. Thus A/B/C/E fail and D matches.

B - Incorrect: "strlen(msg) would return 24" does not match the rule or result derived here. The module supports char-array string initialization and output. Printing msg displays its literal This

is the final exam for C++. Its length is 28 characters, not 24. strcpy(destination,source) copies Hurray into msg, not the reverse. Thus A/B/C/E fail and D matches.

C - Incorrect: "strcpy(msg, "Hurray") would assign the content of msg to Hurray" does not match the rule or result derived here. The module supports char-array string initialization and output. Printing msg displays its literal This is the final exam for C++. Its length is 28 characters, not 24. strcpy(destination,source) copies Hurray into msg, not the reverse. Thus A/B/C/E fail and D matches.

D - Correct: "cout<<msg ; would display: This is the final exam for C++" matches the derived result. The module supports char-array string initialization and output. Printing msg displays its literal This is the final exam for C++. Its length is 28 characters, not 24. strcpy(destination,source) copies Hurray into msg, not the reverse. Thus A/B/C/E fail and D matches.

E - Incorrect: "None of the answers are correct" does not match the rule or result derived here. The module supports char-array string initialization and output. Printing msg displays its literal This is the final exam for C++. Its length is 28 characters, not 24. strcpy(destination,source) copies Hurray into msg, not the reverse. Thus A/B/C/E fail and D matches.

Primary note proof

Study-note section CP24

A C string is a character array ending in '\0'. "Hello" needs six char positions; strlen counts five visible characters and excludes the terminator. sizeof a fixed char array reports its allocated bytes, which can exceed strlen+1. cout<<a prints a terminated char sequence, not its numeric address.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 22 | lines 22-38

String and pointer String: - is an ordered sequence of characters - Sequence of character in which the last character is the null character „\0‟ - The null character indicates the end of the string - C++ provides following two types of string representations: o The C-style character string. o The string class type introduced with Standard C++. NB: →If a string has n characters, then it requires an n+1 element array (at least) to store it. →In order to use the string data type, the C++ string header <string> must be included at the top of the program The C-Style Character String - originated within the C language and continues to be supported within C++ - array of characters which is terminated by a null character '\0' Example: char status[5]={„G‟,‟O‟,‟O‟,‟D‟,‟\0‟};//char status[]=”Good”; cout<<status;

Computer Programming module | PDF page 23 | lines 2-8

- C++ supports a wide range of functions that manipulate null-terminated strings: strcpy(s1, s2); Copies string s2 into string s1 strcat(s1, s2); Concatenates string s2 onto the end of string s1 strlen(s1); Returns the length of string s1. strcmp(s1, s2); Returns 0 if s1 and s2 are the same; less than 0 if s1<s2; greater than 0 if s1>s2. strchr(s1, ch); Returns a pointer to the first occurrence of character ch in string s1. strstr(s1, s2); Returns a pointer to the first occurrence of string s2 in string s1.

Computer Programming module | PDF page 6 | lines 4-15

> 6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Mock exam 3 · Question 7

Question 82 | Q01576

Given the structure below, assume ptr1 and ptr2 point to separate live house objects. Which statement copies the rooms value from the first object to the second?
struct house { double price; int rooms; };
house *ptr1, *ptr2;

A. Cout<<ptr1<< ptr2;

B. ptr2->rooms = ptr1->rooms;

C. ptr1 < ptr2;

D. ptr1 = ptr1 + ptr2;

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: B. ptr2->rooms = ptr1->rooms;

Each pointer must refer to a live object before its members are accessed. The -> operator selects a member through a pointer, so assigning ptr1->rooms to ptr2->rooms copies the first object’s integer member into the second. A is an output expression with a misspelled stream name, C compares pointers rather than copying a member, and D tries to add two pointers, which is not permitted pointer arithmetic.

Why each choice is right or wrong

A - Incorrect: This output expression does not copy rooms and uses undeclared Cout.

B - Correct: Both -> expressions select rooms, and assignment copies from ptr1 to ptr2.

C - Incorrect: A pointer comparison does not copy a member value.

D - Incorrect: Adding two pointers is not valid pointer arithmetic.

Primary note proof

Study-note section CP27

> A struct groups named members, possibly of different types. struct House{double price; int rooms;}; defines a type; House h{150000,3}; creates an object. h.rooms uses a dot. House* p=&h;

p->rooms is equivalent to (*p).rooms. The pointer must refer to a live object before member access. A type definition alone does not create an ordinary instance.

Study-note section CP26

Pointer addition p+q is invalid; subtracting pointers within the same array gives an element distance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 38 | lines 10-16

Pointer to structure Pointers to structures may be declared as follow: struct staructuretage *pinternname; eg Employe *emp; When using a pointer to a struct, member access can be achieved with the `.' operator, but can not look graceful in expression Example: (*emp).id Equivalently, the ->' operator can be used; for example: emp->id

Computer Programming module | PDF page 26 | lines 2-22

Pointer - A pointer is a variable that holds a memory address of another variable. - - Its content is a memory address - A pointer is simply the address of a memory location and provides an indirect way of accessing data in memory. - It is defined to „point to“ data of a specific type. - Because a pointer variable holds the address of another piece of data, it "points to" the data Reference operator (&) and Dereference operator (*) &: read as address of - Use address operator & to get address of a variable *: read as value pointed by Declaring variables of pointer types Syntax: type *variable_name; For example: int *ptr1; //pointertoanint double*number; char*ptr2; //pointertoachar The value of a pointer variable is the address to which it points. For example, given the definitions int num; wecanwrite: ptr1=#

Question sources

Mock exam 3 · Question 8

Question 83 | Q01577

Under C++17, which option best summarizes whether initialization forms A-C are valid? Treat each form as a separate declaration; 25.00 is exactly representable as float.

A. struct Time { int h = 06, m = 20; float s = 25.00; };

B. struct Time { int h, m; float s;}t = {06, 20, 25.00};

C. struct Time { int h, m; float s; }; Time t = {06, 20, 25.00};

D. All of them are correct

E. None of the choices are correct

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer D: D. All of them are correct

C++17 permits default member initializers, so A's in-class values are allowed. B defines an aggregate object alongside its type, while C defines the type and then aggregate-initializes an object. The specified exactly representable floating constant is permitted here's narrowing exception. All three forms are consequently valid. The question asks for the complete summary, making D the best selection rather than any one individual example.

Why each choice is right or wrong

A - Incorrect: A is valid but alone does not summarize all three forms.

B - Incorrect: B is valid but describes only one of the three valid forms.

C - Incorrect: C is valid but is an incomplete response to the requested summary.

D - Correct: All three forms are valid under the specified C++17 assumptions.

E - Incorrect: The existence of valid forms A-C makes the none-correct summary false.

Primary note proof

Study-note section CP27

> Aggregate initialization assigns members in declaration order. Modern C++ permits default member initializers such as int rooms=3;. C++11 list initialization rejects narrowing; a constant floating value exactly representable in the target floating type can be permitted, so 25.00 to float is not necessarily narrowing. Use 25.0f for clarity and check the language version. Structures can be copied, passed and returned; a structure cannot contain a full instance of itself, but may contain a pointer to its own type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 35 | lines 37-44

> Initialize value #include<iostream> using namespace std; struct student{ string name; int age; double score; }st={"Abebe",19,24};

Question sources

Mock exam 3 · Question 9

Question 84 | Q01615

If we have object from ofstream class, then default mode of opening the file is _____ .

A. ios::in

B. ios::out

C. ios::out|ios::trunk

D. ios::in|ios::trunc

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: B. ios::out

The ofstream constructor's default mode argument is ios::out. Its usual output-only opening behavior also creates or truncates the file, but that effect does not change the spelling of the default argument. ios::in requests input instead. The printed ios::trunk is not the valid truncation flag, and an input/truncation combination is not the output stream's default. Thus B is the unique default-mode selection.

Why each choice is right or wrong

A - Incorrect: ios::in requests input rather than the default output mode.

B - Correct: ios::out is the default ofstream mode argument.

C - Incorrect: ios::trunk is misspelled; even out|trunc describes an effect rather than the constructor's default argument.

D - Incorrect: Input combined with truncation is not the default output mode argument.

Primary note proof

Study-note section CP29

> Include <fstream>. ofstream writes, ifstream reads, and fstream can do both. The constructor's default ofstream mode argument is ios::out; its ordinary effect is to create/truncate the file. Distinguish the default mode argument from effective out|trunc behavior. ios::app places each write at the end; ios::ate initially seeks to the end; ios::binary requests binary mode.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 41 | lines 2-9

> • fstream: Stream class to both read and write from/to files. File Modes In C++, for every file operation, exists a specific file mode. These file modes allow us to create, read, write, append or modify a file. The file modes are defined in the class ios. Let's see all these different modes in which we could open a file on disk. Create and Write to a File To create a file, use either the ofstream or fstream class, and specify the name of the file. To write to the file, use the insertion operator (<<).

Computer Programming module | PDF page 40 | lines 16-22

> File and File Management Files are used to store data in storage devices permanently. The fstream library allows us to work with files. To use the fstream library, include both the standard <iostream> AND the <fstream> header file: Input/Output with files • ofstream: Stream class to write on files • ifstream: Stream class to read from files

Question sources

Mock exam 2 · Question 4

Question 85 | Q01616

After the prompt and input of vehicleType=2, what text is printed by the switch body?

```
int vehicleType;
cout << "Enter vehicle type: ";
cin >> vehicleType;
switch (vehicleType) {
case 1:
cout << "Passenger car.";
case 2:
cout << "Bus.";
case 3:
cout << "Truck.";
break;
default:
cout << "Unknown vehicle class!";
}
```

A. Passenger Car.

B. Bus.Truck.

C. Truck

D. Bus.

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: B. Bus.Truck.

Input 2 begins execution at case 2, so Bus. prints first. There is no break after that output, so execution continues through case 3 and prints Truck. before reaching the break. The switch-body output is therefore Bus.Truck. The input prompt is excluded explicitly. The other options either start at the wrong case or omit one of the outputs produced by fall-through.

Why each choice is right or wrong

A - Incorrect: Case 1 is not entered when the value is 2.

B - Correct: Case 2 falls through to case 3 before the break.

C - Incorrect: The matching case prints Bus. before Truck.

D - Incorrect: The absent break after case 2 allows Truck. to print too.

Primary note proof

Study-note section CP13

A traditional C++ switch selects using an integral or enumeration value. case labels are constant expressions. Execution starts at the matching label, then continues through subsequent statements until a break, return or the end. default handles an unmatched value but need not be the last physical label.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 10 | lines 15-26

switch Statement The switch statement provides a way of choosing between a set of alternatives, based on the value of an expression. The general form of the switch statement is: switch (expression) { caseconstant1: statements; ... caseconstantn: statements; default: statements; }

Computer Programming module | PDF page 13 | lines 30-35

Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Question sources

Mock exam 2 · Question 5

Question 86 | Q01617

For the C++ fragment below, assume sizeof(float)==4, byte-addressed memory, a first-element address of 7000, and ordinary array-to-pointer conversion in B. Evaluate each option independently from ptr=num. Which statement is false?
float num[4]={2.0,1.0,3.0,5.5};
float* ptr=num;

A. ptr + 3 is equal to 7012.

B. num is the same as &num[0].

C. The statement cout<<num[3]; will print 5.5.

D. The statement cout<<ptr++; will print 1.0.

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: D. The statement cout<<ptr++; will print 1.0.

Pointer arithmetic advances by elements, not single bytes. With four-byte float elements, ptr+3 has address 7000+12=7012. The array expression ordinarily converts to &num[0], and index 3 selects the initialized value 5.5. Inserting ptr++ into cout displays its old pointer address and then advances it. It does not insert an array element’s value, so the claim that it prints 1.0 is false.

Why each choice is right or wrong

A - Incorrect: Three float elements advance the address by 12 bytes.

B - Incorrect: In the stated conversion context num becomes &num[0].

C - Incorrect: Index 3 contains the fourth initializer, 5.5.

D - Correct: ptr++ supplies the old pointer address, not the element value 1.0.

Primary note proof

Study-note section CP26

> If p points inside an array, p+k advances k elements, or k*sizeof(*p) bytes. Under the explicit assumption sizeof(float)==4 and first address 7000, p+3 has numeric address 7012. cout<<p++ displays the old address then advances p; cout<<*p++ displays the old element then advances p. ++*p changes the element. Pointer addition p+q is invalid; subtracting pointers within the same array gives an element distance.

Study-note section CP25

> An array expression commonly converts to a pointer to its first element, but the array is not itself an assignable pointer variable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 29 | lines 2-10

> Pointer arithmetic - The C++ language allows you to perform integer addition or subtraction operations on pointers. - If ptr points to an integer, ptr + 1 is the address of the next integer in memory after ptr. - ptr - 1 is the address of the previous integer before ptr. Note that ptr + 1 does not return the memory address after ptr, but the memory address of the next object of the type that ptr points to. If ptr points to an integer (assuming 4 bytes), ptr + 3 means 3 integers (12 bytes) after ptr. If ptr points to a char, which is always 1 byte, ptr + 3 means 3 chars (3 bytes) after ptr. When calculating the result of a pointer arithmetic expression, the compiler always multiplies the integer operand by the size of the object being pointed to. This is called scaling.

Computer Programming module | PDF page 31 | lines 2-11

> - Array can bound into a pointer that points to the first element (element 0) of the array. - An array access can always be implicitly converted to a pointer access of the appropriate type. For example, consider these two declarations: int myarray[20]; int *mypointer; Mypointer=myarray;// mypointer=&myarray[0]; →Stores the address of the first element of the array in variable mypointer → After that, mypointer and myarray would be equivalent and would have very similar properties. →The main difference being that mypointer can be assigned a different address, whereas myarray can never be assigned anything, and will always represent the same block of 20 elements of type int.

Computer Programming module | PDF page 6 | lines 4-15

> 6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int

x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Mock exam 2 · Question 6

Question 87 | SM0009

In the conventional native-execution model, which statement about programming-language generations is false?

A. Assembly language is machine independent

B. Machine language can be executed without translation

C. Assembly language should be translated to machine language before execution

D. Fifth-generation languages are historically associated with declarative logic, constraints and AI/expert systems.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: A. Assembly language is machine independent

Assembly is symbolic source tied to an instruction set; it is not machine independent. Before native execution, an assembler translates it into machine instructions. Those machine instructions are already in the target machine’s instruction format. Fifth generation is instead a historical language category associated with declarative logic, constraints and expert systems, rather than a universal hardware-performance requirement. Only A contradicts the specified language and execution model.

Why each choice is right or wrong

A - Correct: Assembly depends on the target instruction set, so this claim is false.

B - Incorrect: Machine instructions do not require assembly-to-machine translation.

C - Incorrect: Symbolic assembly is translated before native execution.

D - Incorrect: This historical association is the fifth-generation description in the note.

Primary note proof

Study-note section CP03

Machine instructions are conventionally first generation; symbolic assembly is second; general high-level languages such as C++ and Java are third. Database query and application generators such as SQL are conventionally fourth generation. Fifth generation is a loose historical category associated with declarative logic, constraints and AI/expert systems. Generations do not determine whether an implementation uses an interpreter.

Study-note section CP04

In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

MOE Computer Science Exit Exam · Question 3-1

National Computer Science Exit Exam · Question 43

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 71

Question 88 | SM0030

For standard built-in C++ arrays declared with an explicit bound, which statement is false?

A. We use [] square brackets at the time of array declaration

B. Size of an array should be constant at the time of array declaration

C. An array is a collection of similar data objects

D. Elements can be accessed only by writing a bracketed index such as a[i].

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: D. Elements can be accessed only by writing a bracketed index such as a[i].

A standard built-in array declaration uses square brackets and a compile-time constant bound. Its elements share one type. Element access is not restricted to bracket syntax: a[i] is equivalent to *(a+i), and a pointer to an element can also be dereferenced. Option D asserts that bracketed indexing is the only access method, which the pointer equivalence directly disproves.

Why each choice is right or wrong

A - Incorrect: An explicit built-in array bound appears inside square brackets.

B - Incorrect: Standard built-in array bounds must be compile-time constants.

C - Incorrect: Built-in arrays contain same-type elements.

D - Correct: Pointer notation can access elements, so bracketed indexing is not the only method.

Primary note proof

Study-note section CP22

> A built-in C++ array has a fixed number of same-type elements in contiguous storage: int a[4]={10,20,30,40};. Valid indices are 0 through 3. a[4] is outside the array; accessing it has undefined behavior. Java checks bounds and throws ArrayIndexOutOfBoundsException. Do not predict a C++ out-of-range value from nearby memory.

Study-note section CP22

> A standard built-in array bound must be a compile-time constant; variable-length local arrays are compiler extensions. int a[5]={1,2,3}; fills the remaining elements with zeros. An uninitialized automatic array of int is different. a[i] equals *(a+i), so elements can be accessed using pointer notation without explicitly writing brackets.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 19 | lines 4-29

> Arrays - an array is a collection of fixed-size number of elements of the same type stored sequentially in memory. - An individual element of an array is identified by its own unique index (or subscript). - The number of elements must be an integer. - The size of the array is referred to as its dimension - Array in C++ is Zero indexed. Declaring of one-dimensional Array Syntax: elementstype arrayname [size]; → You can use constants such as macro definitions or const ints as the sizes of arrays. Example: int num[5]; string fruit[3]; #define numstudents 100 or const int numstudents=100; double Score[numstudents] ; double Score[numstudents] ; Initialization of an Array - Arrays can be initialized at declaration time in two different ways: Syntax: elementstype arrayname[size] = {list each value separated by comma}; E.g int num[5]={1,2,3,4,5}; elementstype arrayname[ ] = { list with any number of values }; E.g int num[]={1,2,3,4,5}; string fruit[]={“Apple”,“orange”, “Mango”}; Accessing Array Elements - Each element could be accessed using an index into the array - When accessing an array, the index given must be a positive integer from 0 to n-1, where n is the dimension of the array.

Computer Programming module | PDF page 31 | lines 31-36

> NB: →Generally, array[n] is the same as *(array + n), where n is an integer. →Considering that int *ptr; int arr[5]; ptr = arr; ptr + 1 is equivalent to &arr[1]; ptr + 2 is equivalent to &arr[2];

Question sources

MOE Computer Science Exit Exam · Question 9-2

National Computer Science Exit Exam · Question 35

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 72

Question 89 | SM0032

What values are printed by this C++ fragment? Ignore the trailing space; assume cout is available.

```
for(int n=1;n<=18;n=n+2){
if(n%7!=0) cout<<n<<" ";
else break;
}
```

A. 3 5 7

B. 1 3 5 7 9 11 13 15 17

C. 1 3 5

D. 1 2 3 4 5 6

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: C. 1 3 5

The loop begins at 1 and adds two after each completed iteration. The values 1, 3 and 5 are not divisible by seven, so each is printed with a following space. At n=7 the condition is false and the else branch executes break, ending the loop before 7 is printed. Later odd values are never reached. The options show value sequences rather than exact trailing whitespace.

Why each choice is right or wrong

A - Incorrect: The loop prints 1 first and stops before printing 7.

B - Incorrect: break prevents 7 and all later odd values from printing.

C - Correct: The three printed values are 1, 3 and 5.

D - Incorrect: Adding two does not visit consecutive integers.

Primary note proof

Study-note section CP16

> For n=1; n<=18; n+=2, print n unless n%7==0 then break: output is 1 3 5. A while(a>b) with fixed a=6,b=5 never changes its true condition. In data[10]={5,10,...,50}, i=0, advancing i only if data[i]%10==0 causes no progress because data[0]=5. It prints nothing and loops indefinitely; changing <=10 to <10 alone does not repair progress.

Study-note section CP15

break exits the nearest enclosing loop or switch. continue skips the rest of the current loop iteration: a for loop then runs its update; a while loop goes directly to its condition. return leaves the current function and may supply a result. In main it ends the program normally.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 19-28

The for loop. Its format is: for (initialization; condition; increase) statement; 1, initialization is executed. Generally, it is a initial value setting for a counter variable. This is executed only once 2, condition is checked, if it is true the loop continues, otherwise the loop finishes and statement is skipped. 3, statement is executed. As usual, it can be either a single instruction or a block of instructions enclosed within curly brackets { }. 4, finally, whatever is specified in the increase field is executed and the loop gets back to step 2.

Computer Programming module | PDF page 13 | lines 30-35

Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

MOE Computer Science Exit Exam · Question 10-1

Question 90 | SM0055

In a C++ translation unit, int requests=0; is declared at namespace scope before the function definitions. Functions can use it wherever that declaration is visible. Which listed term describes its scope category rather than its storage duration?

A. Static

B. Automatic

C. Local

D. Global

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: D. Global

The declaration is at namespace scope, outside every function, so the variable has global scope in the terminology used here. Its declaration must still be visible where it is used, and a local name can shadow it. Static and automatic concern storage properties rather than identifying this outside-function scope category. The question therefore selects Global without claiming unrestricted visibility across every file.

Why each choice is right or wrong

A - Incorrect: Static storage duration does not uniquely identify scope; a static local can still have block scope.

B - Incorrect: Automatic describes storage duration and does not name the namespace-scope category.

C - Incorrect: A local name is restricted to its function or block.

D - Correct: Global is the category for this namespace-scope declaration, subject to visibility and shadowing.

Primary note proof

Study-note section CP20

> Scope is where a name can be used; lifetime is when its object exists. A local name is restricted to its block; an outer name may be shadowed. A namespace/global variable is available where its declaration is visible, not magically in every file. C++ ::name selects a global name hidden by a local one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 17 | lines 14-29

> Scope of variables Scope of variables defined as the extent of the program code within which the variable can be accessed or declared or worked with mainly two types of variable scopes: local

variable: - Variables defined within a function or block are said to be local to those functions - Declared inside a block{} Global Variable - variable that can be accessed from any part of the program. - They are available throughout the life time of a program. - They are declared at the top of the program outside all of the functions or blocks. NB: - Usually when two variables with same name are defined then the compiler produces a compile time error. But if the variables are defined in different scopes, then the compiler allows it. - Whenever there is a local variable defined with same name as that of a global variable then the compiler will give precedence to the local variable

Question sources

MOE Computer Science Exit Exam · Question 16-2

National Computer Science Exit Exam · Question 60

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 74

Question 91 | SM0088

Which one of the following is NOT an example of scripting languages?

A. Python

B. Perl

C. PHP

D. C++

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: C++.

CP03 identifies Python, Perl and PHP as commonly called scripting languages and C++ as conventionally compiled. This supports C++ as the intended exception. The classification describes common use rather than an absolute prohibition on interpreting C++ or compiling languages normally used for scripting.

Why each choice is right or wrong

A - Incorrect: Python is explicitly included among the note’s commonly called scripting languages.

B - Incorrect: Perl is explicitly included among the note’s commonly called scripting languages.

C - Incorrect: PHP is explicitly included among the note’s commonly called scripting languages.

D - Correct: C++ is the conventionally compiled language contrasted with the three scripting-language examples.

Primary note proof

Study-note section CP03

Machine and assembly code depend on an instruction set. High-level source is easier to read and often portable after recompilation, provided it avoids platform assumptions. People can write both assembly and high-level programs. Python, Perl and PHP are commonly called scripting languages; C++ is conventionally compiled. These labels describe common use, not an absolute technical prohibition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Computer Programming module | PDF page 4 | lines 4-19

programming paradigm: provides the programmer's view of code execution. The most influential paradigms are: Procedural Programming Languages Procedural programming specifies a list of operations that the program must complete to reach the desired state. Each program has a starting state, a list of operations to complete, and an ending point. This approach is also known as imperative programming. Structured Programming Languages Structured programming is a special type of procedural programming. It provides additional tools to manage the problems that larger programs were creating. Structured programming requires that programmers break program structure into small pieces of code that are easily understood. One of the well-known features of structural programming is that it does not allow the use of the GOTO statement. It is often associated with a "top-down" approach to design. Object-Oriented Programming Languages Object-oriented programming is one the newest and most powerful paradigms. In object- oriented programs, the designer specifies both the data structures and the types of operations that can be applied to those data structures.

Question sources

MOE Computer Science Exit Exam · Question 25-1

National Computer Science Exit Exam · Question 12

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 28

Question 92 | SM0092

What is the output of the following C++ fragment code? int a, b; a = 13; b = 9; while((20 && 0) && (a > b)) { cout << "Plants are our life"; }

A. Displays 'Plants are our life' infinitely

B. No message will be displayed

C. Displays 'Plants are our life' 20 times

D. Displays 'Plants are Our Life' 4 times

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: No message will be displayed

20 is logically true but 0 is false, so 20&&0 is false. Its conjunction with a>b remains false even though 13>9. The while body is never entered, hence no message; none of the positive repetition counts or infinite-output option fits. Module support: PDF p. 11, PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "Displays 'Plants are our life' infinitely" does not match the rule or result derived here. 20 is logically true but 0 is false, so 20&&0 is false. Its conjunction with a>b remains false even though 13>9. The while body is never entered, hence no message; none of the positive repetition counts or infinite-output option fits.

B - Correct: "No message will be displayed" matches the derived result. 20 is logically true but 0 is false, so 20&&0 is false. Its conjunction with a>b remains false even though 13>9. The while body is never entered, hence no message; none of the positive repetition counts or infinite-output option fits.

C - Incorrect: "Displays 'Plants are our life' 20 times" does not match the rule or result derived here. 20 is logically true but 0 is false, so 20&&0 is false. Its conjunction with a>b remains false even though 13>9. The while body is never entered, hence no message; none of the positive repetition counts or infinite-output option fits.

D - Incorrect: "Displays 'Plants are Our Life' 4 times" does not match the rule or result derived here. 20 is logically true but 0 is false, so 20&&0 is false. Its conjunction with a>b remains false even though 13>9. The while body is never entered, hence no message; none of the positive repetition counts or infinite-output option fits.

Primary note proof

Study-note section CP10

> Relational and equality operators test values; && and || combine truth values; ! negates. In C++, zero converts to false and a nonzero number to true. Streams display bool as 0 or 1 unless boolalpha is set. With a=3,b=5, cout<<(a==b) prints 0.

Study-note section CP14

while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 11 | lines 5-13

Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

MOE Computer Science Exit Exam · Question 26-2

National Computer Science Exit Exam · Question 85

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 78

Question 93 | SM0204

In C++, what does the cout object typically represent?

A. Standard input stream

B. Keyboard input buffer

C. Error output stream

D. Standard output stream

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Standard output stream

The module explicitly calls cout the standard output stream (p.6). Standard input is cin, and keyboard buffer/error stream do not name the cout role. Module support: PDF p. 6.

Why each choice is right or wrong

A - Incorrect: "Standard input stream" does not match the rule or result derived here. The module explicitly calls cout the standard output stream (p.6). Standard input is cin, and keyboard buffer/error stream do not name the cout role.

B - Incorrect: "Keyboard input buffer" does not match the rule or result derived here. The module explicitly calls cout the standard output stream (p.6). Standard input is cin, and keyboard buffer/error stream do not name the cout role.

C - Incorrect: "Error output stream" does not match the rule or result derived here. The module explicitly calls cout the standard output stream (p.6). Standard input is cin, and keyboard buffer/error stream do not name the cout role.

D - Correct: "Standard output stream" matches the derived result. The module explicitly calls cout the standard output stream (p.6). Standard input is cin, and keyboard buffer/error stream do not name the cout role.

Primary note proof

Study-note section CP28

> Include <iostream> and qualify std::cin/std::cout or introduce the names explicitly. cout is the standard output stream; << inserts values. cin is standard input; >> extracts formatted values. std::endl inserts a newline and flushes. C++ names are case sensitive: Cout, Int and std: :cout are not equivalent valid spellings of cout, int and std::cout.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 6 | lines 4-15

> 6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Computer Science Exit Exam · Sene 2016 · Question 1-4

Computer Science Exit Exam · Sene 2016 · Question 19-2

Computer Science Exit Exam practice collection · Question 93

Question 94 | SM0209

What is the term for a named block of code that performs a specific task?

A. Condition

B. Variable

C. Function

D. Loop

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Function

A function is defined as a block of code performing a specific task and called from a program point (p.14). A condition supplies a test, a variable stores data, and a loop provides repetition rather than the named task unit. Module support: PDF p. 14.

Why each choice is right or wrong

A - Incorrect: "Condition" does not match the rule or result derived here. A function is defined as a block of code performing a specific task and called from a program point (p.14). A condition supplies a test, a variable stores data, and a loop provides repetition rather than the named task unit.

B - Incorrect: "Variable" does not match the rule or result derived here. A function is defined as a block of code performing a specific task and called from a program point (p.14). A condition supplies a test, a variable stores data, and a loop provides repetition rather than the named task unit.

C - Correct: "Function" matches the derived result. A function is defined as a block of code performing a specific task and called from a program point (p.14). A condition supplies a test, a variable stores data, and a loop provides repetition rather than the named task unit.

D - Incorrect: "Loop" does not match the rule or result derived here. A function is defined as a block of code performing a specific task and called from a program point (p.14). A condition supplies a test, a variable stores data, and a loop provides repetition rather than the named task unit.

Primary note proof

Study-note section CP17

> A function groups a named task. C++ definition: int add(int a,int b){return a+b;}. Call it with add(x,y);, without parameter type declarations at the call site. Parameters are names in the definition; arguments are supplied expressions or values. A void function produces no return value but can have side effects and return; to exit early.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 14 | lines 21-33

> Function A function is a group of statements that structure our programs in a more modular way and executed when it is called from some point of the program. It is a block of code that performs a specific task. It is important to Dividing a complex problem into smaller chunks makes our program easy to understand and reusable. When we need function? - When you need to repeat the same process over and over in a program. - The function can be called many times but appears in the code once Function - Categories C++ functions can be divided into 2 categories: • pre-defined (standard function) / library function - Which is the definitions have been written and it is ready to be used. - User needs to include pre-defined header file (i.e., math.h, time.h,cmath) To use it: FunctionName (value) e.g., sqrt (4), floor (4.3)

Question sources

Computer Science Exit Exam · Sene 2016 · Question 3-2

Computer Science Exit Exam practice collection · Question 89

Question 95 | SM0217

What does the if-else statement allow you to do in C++?

A. Declare and define functions

B. Allocate memory dynamically

C. Execute a block of code only if a condition is true, otherwise execute another block

D. Repeat a block of code multiple times

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Execute a block of code only if a condition is true, otherwise execute another block

The p.9 if-else rule evaluates a condition and executes statement1 when true, otherwise statement2. This matches C. It is not a declaration, allocation mechanism or repetition construct. Module support: PDF p. 9.

Why each choice is right or wrong

A - Incorrect: "Declare and define functions" does not match the rule or result derived here. The p.9 if-else rule evaluates a condition and executes statement1 when true, otherwise statement2. This matches C. It is not a declaration, allocation mechanism or repetition construct.

B - Incorrect: "Allocate memory dynamically" does not match the rule or result derived here. The p.9 if-else rule evaluates a condition and executes statement1 when true, otherwise statement2. This matches C. It is not a declaration, allocation mechanism or repetition construct.

C - Correct: "Execute a block of code only if a condition is true, otherwise execute another block" matches the derived result. The p.9 if-else rule evaluates a condition and executes statement1

when true, otherwise statement2. This matches C. It is not a declaration, allocation mechanism or repetition construct.

D - Incorrect: "Repeat a block of code multiple times" does not match the rule or result derived here. The p.9 if-else rule evaluates a condition and executes statement1 when true, otherwise statement2. This matches C. It is not a declaration, allocation mechanism or repetition construct.

Primary note proof

Study-note section CP12

> A standalone if may execute independently of the following if. An else binds to its nearest unmatched if; an else-if ladder executes the first true branch only. In nested if(c1){if(c2) S1; else S2;}, S2 requires c1 true and c2 false. Braces determine grouping.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 9 | lines 9-21

> Conditional Statements A. The if Statement Syntax: if (expression) statement; First expression is evaluated. If the outcome is nonzero (true) then statement is executed. Otherwise, nothing happens. if-else statement Syntax: if (expression) statement1; else statement2; First expression is evaluated. If the outcome is nonzero (true) then statement1 is executed. Otherwise, statement2 is executed.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 6-1

Computer Science Exit Exam practice collection · Question 90

Question 96 | SM0231

In computer programming, what is the purpose of control structures such as loops and conditional statements?

A. To control the flow of program execution

B. To perform mathematical operations

C. To define the data types used in a program

D. To allocate memory for variables

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: To control the flow of program execution

The module says flow control determines what executes during a run; branches choose alternative paths and loops repeat computations (p.9). Thus these structures control execution flow, rather than merely defining types, allocating memory or computing arithmetic. Module support: PDF p. 9.

Why each choice is right or wrong

A - Correct: "To control the flow of program execution" matches the derived result. The module says flow control determines what executes during a run; branches choose alternative paths and loops repeat computations (p.9). Thus these structures control execution flow, rather than merely defining types, allocating memory or computing arithmetic.

B - Incorrect: "To perform mathematical operations" does not match the rule or result derived here. The module says flow control determines what executes during a run; branches choose alternative paths and loops repeat computations (p.9). Thus these structures control execution flow, rather than merely defining types, allocating memory or computing arithmetic.

C - Incorrect: "To define the data types used in a program" does not match the rule or result derived here. The module says flow control determines what executes during a run; branches choose alternative paths and loops repeat computations (p.9). Thus these structures control execution flow, rather than merely defining types, allocating memory or computing arithmetic.

D - Incorrect: "To allocate memory for variables" does not match the rule or result derived here. The module says flow control determines what executes during a run; branches choose alternative paths and loops repeat computations (p.9). Thus these structures control execution flow, rather than merely defining types, allocating memory or computing arithmetic.

Primary note proof

Study-note section CP12

> A standalone if may execute independently of the following if. An else binds to its nearest unmatched if; an else-if ladder executes the first true branch only. In nested if(c1){if(c2) S1; else S2;}, S2 requires c1 true and c2 false. Braces determine grouping.

Study-note section CP14

> while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 9 | lines 2-8

> Control Statements Control statements or flow control determines what is executed during a run and what is not, therefore affecting the overall outcome of the program. C++ provides different forms of statements for different purposes. Declaration statements are used for defining variables. Assignment-like statements are used for simple, algebraic computations. Branching statements are used for specifying alternate paths of execution, depending on the outcome of a

logical condition. Loop statements are used for specifying computations, which need to be repeated until a certain logical condition is satisfied

Question sources

Computer Science Exit Exam · Sene 2016 · Question 10-2

Computer Science Exit Exam practice collection · Question 91

Question 97 | SM0236

When encountering a syntax error in code, what is the first step you should take to resolve it?

A. Rewrite the entire

B. Ignore the error and continue execution

C. Usea debugger tool

D. Check the code for typos or incorrect syntax

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer D: Check the nearby code for typos and incorrect syntax.

CP32 recommends reading the first useful compiler diagnostic and inspecting nearby tokens and declarations before repairing and rebuilding. Among the offered actions, checking for typos or incorrect syntax matches that early repair step. Ignoring the error cannot fix translation, and a debugger is most useful once a runnable program exists.

Why each choice is right or wrong

A - Incorrect: Rewriting everything is unnecessary; the note recommends diagnosing and repairing the relevant local code.

B - Incorrect: Ignoring a syntax error neither corrects grammar nor establishes that an executable can be built.

C - Incorrect: A debugger is useful after a runnable program exists; compiler diagnostics guide syntax repair first.

D - Correct: Inspecting tokens and declarations for syntax mistakes matches the first useful action among these options.

Primary note proof

Study-note section CP32

A syntax error violates grammar; a type error can be syntactically well formed but rejected during semantic checking. A runtime error occurs during execution; a logic error gives the wrong behavior. A bug is a defect; debugging locates and corrects it. Read the first useful compiler diagnostic, inspect nearby tokens and declarations, repair and rebuild; a debugger is useful after a runnable program exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Computer Science Exit Exam · Sene 2016 · Question 12-1

Question 98 | SM0276

Which C++ statement is used to terminate a loop prematurely and transfer control to the next iteration of the loop?

A. break

B. exit

C. return

D. continue

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: continue

The phrase transfer control to the next iteration describes continue, which skips remaining statements of the current iteration and preserves the loop (p.13). break exits the loop entirely; return leaves a function; exit is not the loop-iteration statement. Module support: PDF p. 13.

Why each choice is right or wrong

A - Incorrect: "break" does not match the rule or result derived here. The phrase transfer control to the next iteration describes continue, which skips remaining statements of the current iteration and preserves the loop (p.13). break exits the loop entirely; return leaves a function; exit is not the loop-iteration statement.

B - Incorrect: "exit" does not match the rule or result derived here. The phrase transfer control to the next iteration describes continue, which skips remaining statements of the current iteration and preserves the loop (p.13). break exits the loop entirely; return leaves a function; exit is not the loop-iteration statement.

C - Incorrect: "return" does not match the rule or result derived here. The phrase transfer control to the next iteration describes continue, which skips remaining statements of the current

iteration and preserves the loop (p.13). break exits the loop entirely; return leaves a function; exit is not the loop-iteration statement.

D - Correct: "continue" matches the derived result. The phrase transfer control to the next iteration describes continue, which skips remaining statements of the current iteration and preserves the loop (p.13). break exits the loop entirely; return leaves a function; exit is not the loop-iteration statement.

Primary note proof

Study-note section CP15

> break exits the nearest enclosing loop or switch. continue skips the rest of the current loop iteration: a for loop then runs its update; a while loop goes directly to its condition. return leaves the current function and may supply a result. In main it ends the program normally.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 13 | lines 30-35

> Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 25-1

Computer Science Exit Exam practice collection · Question 94

Question 99 | SM0288

Which is a desirable performance quality rather than a necessary defining requirement for a useful algorithm with a specified result on every legal input?

A. Definiteness

B. Output

C. Finiteness

D. Efficiency

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: D. Efficiency

The note distinguishes the existence of an algorithm from how efficiently it runs. Its instructions must be unambiguous, it must specify a result, and it must terminate for legal inputs. An inefficient procedure can still satisfy all these defining requirements and correctly solve its problem. Efficiency is therefore desirable and measurable but is the exception when the question asks specifically about necessary defining requirements.

Why each choice is right or wrong

A - Incorrect: Definiteness requires unambiguous instructions.

B - Incorrect: A useful algorithm specifies a result.

C - Incorrect: Finiteness requires termination for legal inputs.

D - Correct: Efficiency is desirable but an inefficient terminating algorithm still qualifies.

Primary note proof

Study-note section CP02

> An algorithm gives precise, executable steps and terminates for every legal input. Definiteness means each step has an unambiguous meaning; effectiveness means a step can actually be performed. Inputs may be absent, but a useful algorithm specifies a result. Correctness means the result meets the specification. Efficiency is desirable and measurable, but an inefficient terminating algorithm is still an algorithm. Efficiency is also a desirable property of an algorithm.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 19-26

> Algorithm is an ordered set of unambiguous, executable steps, defining a terminating process. Important Properties of Algorithms - Correct: always returns the desired output for all legal instances of the problem. - Unambiguous: Each step must be clearly defined, having only one interpretation. - Precise: simple but efficient - Efficient: can be measured in terms of Time and Space - Finiteness: the execution of the algorithm must terminate in a finite amount of time for each possible set of input data/must have an end.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 29-1

Computer Science Exit Exam practice collection · Question 119

Question 100 | SS049

What is the output of the following C++ fragment code? for(int n = 1; n <= 18; n = n + 2) { if(n % 7 != 0) cout << n << " "; else break; }

A. 3 5 7

B. 1 3 5 7 9 11 13 15 17

C. 1 3 5

D. 1 2 3 4 5 6

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer C: 1 3 5

The loop prints n=1,3,5 because each has nonzero remainder modulo 7. At n=7 the remainder is zero, so break exits before printing 7. A includes a forbidden 7 and omits 1; B ignores break; D ignores the +2 update. Module support: PDF p. 12, PDF p. 13, PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "3 5 7" does not match the rule or result derived here. The loop prints n=1,3,5 because each has nonzero remainder modulo 7. At n=7 the remainder is zero, so break exits before printing 7. A includes a forbidden 7 and omits 1; B ignores break; D ignores the +2 update.

B - Incorrect: "1 3 5 7 9 11 13 15 17" does not match the rule or result derived here. The loop prints n=1,3,5 because each has nonzero remainder modulo 7. At n=7 the remainder is zero, so break exits before printing 7. A includes a forbidden 7 and omits 1; B ignores break; D ignores the +2 update.

C - Correct: "1 3 5" matches the derived result. The loop prints n=1,3,5 because each has nonzero remainder modulo 7. At n=7 the remainder is zero, so break exits before printing 7. A includes a forbidden 7 and omits 1; B ignores break; D ignores the +2 update.

D - Incorrect: "1 2 3 4 5 6" does not match the rule or result derived here. The loop prints n=1,3,5 because each has nonzero remainder modulo 7. At n=7 the remainder is zero, so break exits before printing 7. A includes a forbidden 7 and omits 1; B ignores break; D ignores the +2 update.

Primary note proof

Study-note section CP16

> For n=1; n<=18; n+=2, print n unless n%7==0 then break: output is 1 3 5. A while(a>b) with fixed a=6,b=5 never changes its true condition. In data[10]={5,10,...,50}, i=0, advancing i only if data[i]%10==0 causes no progress because data[0]=5. It prints nothing and loops indefinitely; changing <=10 to <10 alone does not repair progress.

Study-note section CP15

break exits the nearest enclosing loop or switch. continue skips the rest of the current loop iteration: a for loop then runs its update; a while loop goes directly to its condition. return leaves the current function and may supply a result. In main it ends the program normally.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 19-28

The for loop. Its format is: for (initialization; condition; increase) statement; 1, initialization is executed. Generally, it is a initial value setting for a counter variable. This is executed only once 2, condition is checked, if it is true the loop continues, otherwise the loop finishes and statement is skipped. 3, statement is executed. As usual, it can be either a single instruction or a block of instructions enclosed within curly brackets { }. 4, finally, whatever is specified in the increase field is executed and the loop gets back to step 2.

Computer Programming module | PDF page 13 | lines 30-35

Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

National Computer Science Exit Exam · Question 48

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 73

Question 101 | SS160

For input b=4 and e=3, what numeric result is printed after the prompt by this repaired C++ program?

```
#include <iostream>
using namespace std;
int main(){
int b,e,r=1;
cout<<"Enter b and e";
cin>>b>>e;
for(int i=1;i<=e;i++){r=r*b;}
cout<<r;
return 0;
}
```

A. 81

B. 256

C. 12

D. 64

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer D: D. 64

The accumulator r starts at one. With exponent e=3, the for loop executes for i=1, 2 and 3, multiplying by the base b=4 on each pass. Its successive values are 4, 16 and 64. The final insertion therefore prints 64 after the prompt. This is iterative exponentiation, rather than multiplying the base by the exponent or reversing their roles.

Why each choice is right or wrong

A - Incorrect: 81 is 3 raised to 4, reversing base and exponent.

B - Incorrect: 256 would require four multiplications by 4.

C - Incorrect: 12 multiplies base by exponent rather than computing the power.

D - Correct: Three multiplications by 4 starting at 1 give 64.

Primary note proof

Study-note section CP30

> C++ has no built-in exponentiation operator: ^ is bitwise XOR and ** is not a power operator. To compute b^e for a nonnegative integer e, start r=1 and repeat r*=b exactly e times; b=4,e=3 gives 64. Large powers and factorials can overflow fixed-width types; specify a valid input range.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 19-28

The for loop. Its format is: for (initialization; condition; increase) statement; 1, initialization is executed. Generally, it is a initial value setting for a counter variable. This is executed only once 2, condition is checked, if it is true the loop continues, otherwise the loop finishes and statement is skipped. 3, statement is executed. As usual, it can be either a single instruction or a block of instructions enclosed within curly brackets { }. 4, finally, whatever is specified in the increase field is executed and the loop gets back to step 2.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science model exam collection · Question 67

Question 102 | SS167

A program reads exactly 50 numbers and adds only positive values to a total. Which listed construct best expresses the known repetition count, while a separate if statement handles the positivity test?

A. for

B. break

C. jump

D. continue

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: A. for

The task has a counting responsibility and a filtering responsibility. A for loop naturally expresses the fixed total of 50 input attempts through its initialization, condition and update. The question assigns the positivity decision to a separate if, so it asks specifically about counting. continue can skip one contribution but cannot supply the repetition count; break would stop early, and jump is not a specific count-controlled construct.

Why each choice is right or wrong

A - Correct: for naturally expresses exactly 50 iterations.

B - Incorrect: break could stop processing before all 50 inputs.

C - Incorrect: Jump does not specify a count-controlled loop.

D - Incorrect: continue skips the rest of an iteration but does not establish 50 repetitions.

Primary note proof

Study-note section CP23

> For a sum, start total=0 and add each element once. For a maximum/minimum, initialize from a[0] only after confirming the array is nonempty. For 50 inputs where nonpositive values are skipped, use a count-controlled loop and either if(x>0) total+=x; or continue for x<=0. These constructs solve different parts of the task.

Study-note section CP14

> for(initialization; condition; update) executes initialization once, then condition → body → update → condition. A missing condition in for(;;) acts as true. A for loop is convenient for a known count but can also be condition controlled. To print indices 0 through n-1, use i=0; i<n; ++i. Starting i=0 with i<5 and i+=2 prints 0 2 4.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 19-28

> The for loop. Its format is: for (initialization; condition; increase) statement; 1, initialization is executed. Generally, it is a initial value setting for a counter variable. This is executed only once 2, condition is checked, if it is true the loop continues, otherwise the loop finishes and statement is skipped. 3, statement is executed. As usual, it can be either a single instruction or a block of instructions enclosed within curly brackets { }. 4, finally, whatever is specified in the increase field is executed and the loop gets back to step 2.

Computer Programming module | PDF page 13 | lines 30-35

> Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Question sources

Computer Science model exam collection · Question 74

Question 103 | SS173

Which sequence of array values is printed by this Java code? Ignore each println call's trailing space and line separator.

```
class PrintResult {
public static void main(String[] args){
int[] arr={3,4,5,6,7};
for(int i=0;i<arr.length-2;++i){
System.out.println(arr[i]+" ");
}
}
}
```

A. 3 4 5 6 7

B. 3 4 5 6

C. 3 4

D. 3 4 5

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: D. 3 4 5

The initializer contains five values, so arr.length-2 is 3. The loop starts at index zero and continues while i<3, visiting indices 0, 1 and 2. Those positions contain 3, 4 and 5. Each println also ends its line, but the question explicitly asks only for the value sequence.

Why each choice is right or wrong

A - Incorrect: The loop never reaches indices 3 and 4.

B - Incorrect: Four values would require the excluded index 3.

C - Incorrect: Index 2 also satisfies i<3, so a third value prints.

D - Correct: Indices 0,1,2 contain 3,4,5 in that order.

Primary note proof

Study-note section CP23

> In Java, length belongs to each array; a traversal ending at arr.length-2 visits fewer elements than a full traversal.

Study-note section CP22

> Valid indices are 0 through 3.

Study-note section CP14

> for(initialization; condition; update) executes initialization once, then condition → body → update → condition.

Study-note section CP-S04

Java System.out.print writes the supplied value without automatically ending the line. System.out.println writes its value and then a line separator. Successive println calls therefore place their values on separate lines, even when each printed expression also contains a trailing space.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 19 | lines 4-29

Arrays - an array is a collection of fixed-size number of elements of the same type stored sequentially in memory. - An individual element of an array is identified by its own unique index (or subscript). - The number of elements must be an integer. - The size of the array is referred to as its dimension - Array in C++ is Zero indexed. Declaring of one-dimensional Array Syntax: elementstype arrayname [size]; → You can use constants such as macro definitions or const ints as the sizes of arrays. Example: int num[5]; string fruit[3]; #define numstudents 100 or const int numstudents=100; double Score[numstudents] ; double Score[numstudents] ; Initialization of an Array - Arrays can be initialized at declaration time in two different ways: Syntax: elementstype arrayname[size] = {list each value separated by comma}; E.g int num[5]={1,2,3,4,5}; elementstype arrayname[ ] = { list with any number of values }; E.g int num[]={1,2,3,4,5}; string fruit[]={“Apple”,“orange”, “Mango”}; Accessing Array Elements - Each element could be accessed using an index into the array - When accessing an array, the index given must be a positive integer from 0 to n-1, where n is the dimension of the array.

Computer Programming module | PDF page 12 | lines 19-28

The for loop. Its format is: for (initialization; condition; increase) statement; 1, initialization is executed. Generally, it is a initial value setting for a counter variable. This is executed only once 2, condition is checked, if it is true the loop continues, otherwise the loop finishes and statement is skipped. 3, statement is executed. As usual, it can be either a single instruction or a block of instructions enclosed within curly brackets { }. 4, finally, whatever is specified in the increase field is executed and the loop gets back to step 2.

Question sources

Computer Science model exam collection · Question 80

Question 104 | SS177

Which activity comes last in fundamental program development in c++ ?

A. Compiling program

B. Writing program

C. Memory allocation for variables

D. Linking

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Linking

Among the listed activities, linking comes after writing/editing and compilation in the module sequence edit source -> compile -> link -> execute (p.4). Memory allocation for variables is not the later named development phase. Execution/loading occurs afterward, but is not an offered option. Module support: PDF p. 4.

Why each choice is right or wrong

A - Incorrect: "Compiling program" does not match the rule or result derived here. Among the listed activities, linking comes after writing/editing and compilation in the module sequence edit source -> compile -> link -> execute (p.4). Memory allocation for variables is not the later named development phase. Execution/loading occurs afterward, but is not an offered option.

B - Incorrect: "Writing program" does not match the rule or result derived here. Among the listed activities, linking comes after writing/editing and compilation in the module sequence edit source -> compile -> link -> execute (p.4). Memory allocation for variables is not the later named development phase. Execution/loading occurs afterward, but is not an offered option.

C - Incorrect: "Memory allocation for variables" does not match the rule or result derived here. Among the listed activities, linking comes after writing/editing and compilation in the module sequence edit source -> compile -> link -> execute (p.4). Memory allocation for variables is not the later named development phase. Execution/loading occurs afterward, but is not an offered option.

D - Correct: "Linking" matches the derived result. Among the listed activities, linking comes after writing/editing and compilation in the module sequence edit source -> compile -> link -> execute (p.4). Memory allocation for variables is not the later named development phase. Execution/loading occurs afterward, but is not an offered option.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started

that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Question sources

Computer Science model exam collection · Question 84

Question 105 | SS179

The correct structure of the for loop statement is:
eooo
ao
5

A. for(condition, initialization, increment/decrement)

B. for(initialization; condition)

C. for(increment/decrement; initialization; condition)

D. for(initialization; condition; increment/decrement)

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: for(initialization; condition; increment/decrement)

The module gives for(initialization; condition; increase) statement (p.12). D has this order and two semicolons. A uses commas and wrong order, B omits the update field and its separator, and C reverses the roles. Module support: PDF p. 12.

Why each choice is right or wrong

A - Incorrect: "for(condition, initialization, increment/decrement)" does not match the rule or result derived here. The module gives for(initialization; condition; increase) statement (p.12). D has this order and two semicolons. A uses commas and wrong order, B omits the update field and its separator, and C reverses the roles.

B - Incorrect: "for(initialization; condition)" does not match the rule or result derived here. The module gives for(initialization; condition; increase) statement (p.12). D has this order and two semicolons. A uses commas and wrong order, B omits the update field and its separator, and C reverses the roles.

C - Incorrect: "for(increment/decrement; initialization; condition)" does not match the rule or result derived here. The module gives for(initialization; condition; increase) statement (p.12). D has this order and two semicolons. A uses commas and wrong order, B omits the update field and its separator, and C reverses the roles.

D - Correct: "for(initialization; condition; increment/decrement)" matches the derived result. The module gives for(initialization; condition; increase) statement (p.12). D has this order and two semicolons. A uses commas and wrong order, B omits the update field and its separator, and C reverses the roles.

Primary note proof

Study-note section CP14

> for(initialization; condition; update) executes initialization once, then condition → body → update → condition. A missing condition in for(;;) acts as true. A for loop is convenient for a known count but can also be condition controlled. To print indices 0 through n-1, use i=0; i<n; ++i. Starting i=0 with i<5 and i+=2 prints 0 2 4.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 19-28

> The for loop. Its format is: for (initialization; condition; increase) statement; 1, initialization is executed. Generally, it is a initial value setting for a counter variable. This is executed only once 2, condition is checked, if it is true the loop continues, otherwise the loop finishes and statement is skipped. 3, statement is executed. As usual, it can be either a single instruction or a block of instructions enclosed within curly brackets { }. 4, finally, whatever is specified in the increase field is executed and the loop gets back to step 2.

Question sources

Computer Science model exam collection · Question 86

Question 106 | SS180

Which one of the following is the correct identifier in c++?

A. 7variable

B. 7VARIABLE

C. $variable

D. variable_1234

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer D: variable_1234

variable_1234 begins with a letter and contains only letters, underscore and digits. A/B begin with 7, violating the initial-character rule, while C uses $, outside the permitted character set. Module support: PDF p. 5.

Why each choice is right or wrong

A - Incorrect: "7variable" does not match the rule or result derived here. variable_1234 begins with a letter and contains only letters, underscore and digits. A/B begin with 7, violating the initial-character rule, while C uses $, outside the permitted character set.

B - Incorrect: "7VARIABLE" does not match the rule or result derived here. variable_1234 begins with a letter and contains only letters, underscore and digits. A/B begin with 7, violating the initial-character rule, while C uses $, outside the permitted character set.

C - Incorrect: "$variable" does not match the rule or result derived here. variable_1234 begins with a letter and contains only letters, underscore and digits. A/B begin with 7, violating the initial-character rule, while C uses $, outside the permitted character set.

D - Correct: "variable_1234" matches the derived result. variable_1234 begins with a letter and contains only letters, underscore and digits. A/B begin with 7, violating the initial-character rule, while C uses $, outside the permitted character set.

Primary note proof

Study-note section CP05

> In the basic ASCII identifier rules, start with a letter or underscore; continue with letters, digits or underscores. Do not use a keyword. C++ is case sensitive: exitExam and ExitExam differ; delete and float are keywords, but FLOAT is a legal identifier. variable_1234 is valid; 4thInput, T-3 and names containing spaces are not single identifiers. Avoid reserved underscore patterns in application code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 2-10

> 1. Keywords (reserved words) - Unique meaning within a C++ program and must not be used for any other purposes. E.g., if, namespace for, switch, continue, float, new, default, break Etc. 2. Identifiers - Name associated with a function or data object and used to refer to that function or data object. An identifier must: o Start with a letter or underscore o Consist only of letters, the digits 0-9, or the underscore symbol _ o Not be a reserved word

Question sources

Computer Science model exam collection · Question 87

Question 107 | AM0245

Which of the following is NOT a programming paradigm?

A. Procedural

B. Object-oriented

C. Functional

D. Conditional

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Conditional is a construct rather than a programming paradigm.

CP02 distinguishes procedural, object oriented and functional programming as ways of organizing computation. A conditional is a selection construct used inside programs rather than a separate paradigm in this classification. Thus the first three options name paradigms, while Conditional is the requested exception.

Why each choice is right or wrong

A - Incorrect: Procedural programming is a paradigm organizing operations into procedures, as described in CP02.

B - Incorrect: Object oriented programming is a paradigm organizing state and behavior in objects.

C - Incorrect: Functional programming is a paradigm emphasizing function evaluation, as described in CP02.

D - Correct: Conditional denotes a selection construct used within paradigms rather than a separate paradigm here.

Primary note proof

Study-note section CP02

> Procedural programming organizes operations in procedures; structured programming combines sequence, selection and iteration; object oriented programming organizes state and behavior in objects; functional programming emphasizes function evaluation. A conditional statement is a construct, not a separate programming paradigm.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 4-19

> programming paradigm: provides the programmer's view of code execution. The most influential paradigms are: Procedural Programming Languages Procedural programming specifies a list of operations that the program must complete to reach the desired state. Each program has a starting state, a list of operations to complete, and an ending point. This approach is also known as imperative programming. Structured Programming Languages Structured programming is a special type of procedural programming. It provides additional tools to manage the problems that larger programs were creating. Structured programming requires that programmers break program structure into small pieces of code that are easily understood. One of the well-known features of structural programming is that it does not allow the use of the GOTO statement. It is often associated with a "top-down" approach to design. Object-Oriented Programming Languages Object-oriented programming is one the newest and most powerful paradigms. In object- oriented programs, the designer specifies both the data structures and the types of operations that can be applied to those data structures.

Question sources

Computer Science MCQ practice collection · Question 245

Question 108 | AM0246

What is the purpose of a compiler?

A. To execute programs

B. To debug programs

C. To translate source code into machine code

D. To optimize program performance

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: To translate source code into machine code

The module describes translating program source before execution, producing object code for linking and later loading. This supports source-to-machine translation as the compiler purpose. Execution belongs later in the pipeline; debugging and optimizing are not its defining role here. Module support: PDF p. 4, PDF p. 3.

Why each choice is right or wrong

A - Incorrect: "To execute programs" does not match the rule or result derived here. The module describes translating program source before execution, producing object code for linking and later loading. This supports source-to-machine translation as the compiler purpose. Execution belongs later in the pipeline; debugging and optimizing are not its defining role here.

B - Incorrect: "To debug programs" does not match the rule or result derived here. The module describes translating program source before execution, producing object code for linking and later loading. This supports source-to-machine translation as the compiler purpose. Execution belongs later in the pipeline; debugging and optimizing are not its defining role here.

C - Correct: "To translate source code into machine code" matches the derived result. The module describes translating program source before execution, producing object code for linking and later loading. This supports source-to-machine translation as the compiler purpose. Execution belongs later in the pipeline; debugging and optimizing are not its defining role here.

D - Incorrect: "To optimize program performance" does not match the rule or result derived here. The module describes translating program source before execution, producing object code for linking and later loading. This supports source-to-machine translation as the compiler purpose. Execution belongs later in the pipeline; debugging and optimizing are not its defining role here.

Primary note proof

Study-note section CP04

In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Computer Programming module | PDF page 3 | lines 8-17

Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Computer Science MCQ practice collection · Question 246

Question 109 | AM0247

What is the main advantage of high-level programming languages?

A. They are easier to read and write than low-level languages.

B. They are faster and more efficient than low-level languages.

C. They provide direct control over hardware resources.

D. They are platform-specific and provide better compatibility.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: High-level languages are easier to read and write.

CP03 says high-level source is easier to read and often portable after recompilation when platform assumptions are avoided. Readability is the advantage stated in A. The note does not promise universally faster execution or greater hardware control, and platform specificity conflicts with the portability advantage rather than explaining it.

Why each choice is right or wrong

A - Correct: Easier source reading and writing is the abstraction benefit directly supported by CP03.

B - Incorrect: The note gives no universal speed or efficiency guarantee for high-level languages over low-level ones.

C - Incorrect: Direct hardware control is not the high-level readability advantage described here.

D - Incorrect: Platform specificity contradicts the potential source portability discussed in CP03.

Primary note proof

Study-note section CP03

> Machine and assembly code depend on an instruction set. High-level source is easier to read and often portable after recompilation, provided it avoids platform assumptions. People can write both assembly and high-level programs. Python, Perl and PHP are commonly called scripting languages; C++ is conventionally compiled. These labels describe common use, not an absolute technical prohibition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Computer Science MCQ practice collection · Question 247

Question 110 | AM0248

Which option most precisely states the information a C++ variable declaration introduces, independently of whether that declaration also defines storage or supplies an initializer?

A. To assign a value to a variable

B. To create a new variable

C. To specify the data type and name of a variable

D. To check the validity of a variable

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer C: C. To specify the data type and name of a variable

Declaration introduces a variable's name and type into the program. Initialization is the separate act of supplying its initial value, and later assignment replaces a value. Describing a declaration merely as creating a variable does not precisely identify its information or distinguish it from an object definition. The question asks for that precise name-and-type purpose, which C directly states.

Why each choice is right or wrong

A - Incorrect: Assigning a value is a different operation from introducing a declaration.

B - Incorrect: Creating an object is less precise and can depend on whether the declaration is also a definition.

C - Correct: A declaration introduces the variable's name and type.

D - Incorrect: A declaration is not primarily a validity check on an existing variable.

Primary note proof

Study-note section CP06

> Write type name; or type name = value; for ordinary variables: int age = 20;. Declaration introduces a name and type; initialization gives its initial value; later assignment replaces a value. a=b copies a scalar value, so later changes to b do not change a. A pointer or Java object reference instead copies an address/reference.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 24-33

> 5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Question sources

Computer Science MCQ practice collection · Question 248

Question 111 | AM0249

What is the purpose of a loop in programming?

A. To perform mathematical calculations

B. To make decisions based on conditions

C. To store and retrieve data

D. To repeat a set of instructions

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: To repeat a set of instructions

Iteration repeats a statement or block, and loops specify computations repeated while/until a logical condition controls continuation (pp.9,11). Arithmetic, decision-making and storage describe different roles, so D is the specific loop purpose. Module support: PDF p. 9, PDF p. 11.

Why each choice is right or wrong

A - Incorrect: "To perform mathematical calculations" does not match the rule or result derived here. Iteration repeats a statement or block, and loops specify computations repeated while/until a logical condition controls continuation (pp.9,11). Arithmetic, decision-making and storage describe different roles, so D is the specific loop purpose.

B - Incorrect: "To make decisions based on conditions" does not match the rule or result derived here. Iteration repeats a statement or block, and loops specify computations repeated while/until a logical condition controls continuation (pp.9,11). Arithmetic, decision-making and storage describe different roles, so D is the specific loop purpose.

C - Incorrect: "To store and retrieve data" does not match the rule or result derived here. Iteration repeats a statement or block, and loops specify computations repeated while/until a logical condition controls continuation (pp.9,11). Arithmetic, decision-making and storage describe different roles, so D is the specific loop purpose.

D - Correct: "To repeat a set of instructions" matches the derived result. Iteration repeats a statement or block, and loops specify computations repeated while/until a logical condition controls continuation (pp.9,11). Arithmetic, decision-making and storage describe different roles, so D is the specific loop purpose.

Primary note proof

Study-note section CP14

while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 9 | lines 2-8

Control Statements Control statements or flow control determines what is executed during a run and what is not, therefore affecting the overall outcome of the program. C++ provides different forms of statements for different purposes. Declaration statements are used for defining variables. Assignment-like statements are used for simple, algebraic computations. Branching statements are used for specifying alternate paths of execution, depending on the outcome of a logical condition. Loop statements are used for specifying computations, which need to be repeated until a certain logical condition is satisfied

Computer Programming module | PDF page 11 | lines 5-13

Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Question sources

Computer Science MCQ practice collection · Question 249

Question 112 | AM0250

Which statement gives the defining relationship between a method and a function?

A. A method is a function associated with a class or object.

B. Functions return a value, while methods do not.

C. Functions can take parameters, while methods cannot.

D. There is no difference; the terms are used interchangeably.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: A. A method is a function associated with a class or object.

The defining feature of a method is its association with a class or object. It is therefore a function with that association, rather than a fundamentally different operation with no return value or no parameters. Both functions and methods can return values or accept arguments according to their definitions. The two terms are related, but method adds the class/object qualification that ordinary function alone does not assert.

Why each choice is right or wrong

A - Correct: A method is a function associated with a class or object.

B - Incorrect: Methods can return values, and functions may return void.

C - Incorrect: Both ordinary functions and methods can take parameters.

D - Incorrect: Method adds a specific class/object association.

Primary note proof

Study-note section CP17

> A method is a function associated with a class or object; methods can return values. Modules group related responsibilities behind interfaces. Reuse reduces duplication; independent functions are easier to understand and test. A library provides reusable components; a framework also supplies an organizing structure and often calls application callbacks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 14 | lines 21-33

> Function A function is a group of statements that structure our programs in a more modular way and executed when it is called from some point of the program. It is a block of code that performs a specific task. It is important to Dividing a complex problem into smaller chunks makes our program easy to understand and reusable. When we need function? - When you need to repeat the same process over and over in a program. - The function can be called many times but appears in the code once Function - Categories C++ functions can be divided into 2 categories: • pre-defined (standard function) / library function - Which is the definitions have been written and it is ready to be used. - User needs to include pre-defined header file (i.e., math.h, time.h,cmath) To use it: FunctionName (value) e.g., sqrt (4), floor (4.3)

Computer Programming module | PDF page 4 | lines 4-19

> programming paradigm: provides the programmer's view of code execution. The most influential paradigms are: Procedural Programming Languages Procedural programming specifies a list of operations that the program must complete to reach the desired state. Each program has a starting state, a list of operations to complete, and an ending point. This approach is also known as imperative programming. Structured Programming Languages Structured programming is a special type of procedural programming. It provides additional tools to manage the problems that larger programs were creating. Structured programming requires that programmers break program structure into small pieces of code that are easily understood. One of the well-known features of structural programming is that it does not allow the use of the

GOTO statement. It is often associated with a "top-down" approach to design. Object-Oriented Programming Languages Object-oriented programming is one the newest and most powerful paradigms. In object- oriented programs, the designer specifies both the data structures and the types of operations that can be applied to those data structures.

Question sources

Computer Science MCQ practice collection · Question 250

Question 113 | AM0251

Which type of error occurs when the program violates the rules of the programming language?

A. Syntax error

B. Runtime error

C. Logic error

D. Semantic error

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: Syntax error

Syntax is defined as rules governing language symbols, whereas semantics deals with meaning of syntactically correct constructs (p.3). Therefore violating the stated language rules is a syntax error. Runtime and logic problems are not the rule category described. Module support: PDF p. 3.

Why each choice is right or wrong

A - Correct: "Syntax error" matches the derived result. Syntax is defined as rules governing language symbols, whereas semantics deals with meaning of syntactically correct constructs (p.3). Therefore violating the stated language rules is a syntax error. Runtime and logic problems are not the rule category described.

B - Incorrect: "Runtime error" does not match the rule or result derived here. Syntax is defined as rules governing language symbols, whereas semantics deals with meaning of syntactically correct constructs (p.3). Therefore violating the stated language rules is a syntax error. Runtime and logic problems are not the rule category described.

C - Incorrect: "Logic error" does not match the rule or result derived here. Syntax is defined as rules governing language symbols, whereas semantics deals with meaning of syntactically correct constructs (p.3). Therefore violating the stated language rules is a syntax error. Runtime and logic problems are not the rule category described.

D - Incorrect: "Semantic error" does not match the rule or result derived here. Syntax is defined as rules governing language symbols, whereas semantics deals with meaning of syntactically correct

constructs (p.3). Therefore violating the stated language rules is a syntax error. Runtime and logic problems are not the rule category described.

Primary note proof

Study-note section CP32

> A syntax error violates grammar; a type error can be syntactically well formed but rejected during semantic checking. A runtime error occurs during execution; a logic error gives the wrong behavior. A bug is a defect; debugging locates and corrects it. Read the first useful compiler diagnostic, inspect nearby tokens and declarations, repair and rebuild; a debugger is useful after a runnable program exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 3 | lines 8-17

> Categories of programming language: - Low-level languages: - - are machine-dependent - They are designed to be run on a particular computer (not portable). Eg. Machine language and Assembly Language High-level languages - Are machine-independent - Can be run on a variety of computers. Programming language require translation source code to machine language before execution. This translation is accomplished by either a compiler or an interpreter. Compilers translate the entire source

Question sources

Computer Science MCQ practice collection · Question 251

Question 114 | AM0252

What is the purpose of debugging in programming?

A. To execute programs

B. To write programs

C. To optimize program performance

D. To find and fix errors in programs

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: Find and fix errors in programs.

A bug is a defect, and CP32 says debugging locates and corrects it. Executing or writing programs describes broader activities; optimizing performance has a different primary aim. Debugging can involve running code and editing it, but those steps serve its specific purpose of finding and fixing errors.

Why each choice is right or wrong

A - Incorrect: Execution can reveal defects, but simply running a program does not define debugging's purpose.

B - Incorrect: Writing programs is implementation; debugging specifically investigates and corrects defects in them.

C - Incorrect: Performance optimization aims at efficiency, whereas debugging aims at correcting erroneous behavior.

D - Correct: Finding and fixing program errors directly matches CP32's stated debugging purpose.

Primary note proof

Study-note section CP32

> A syntax error violates grammar; a type error can be syntactically well formed but rejected during semantic checking. A runtime error occurs during execution; a logic error gives the wrong behavior. A bug is a defect; debugging locates and corrects it. Read the first useful compiler diagnostic, inspect nearby tokens and declarations, repair and rebuild; a debugger is useful after a runnable program exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 2-18

> 1. Introduction to programming Computer programming is the process of writing, testing, debugging/troubleshooting, and maintaining the source code of computer programs. Programs are the instructions that tells the computer what to do .so, writing computer programs means writing instructions, that will make the computer follow and run a program based on those instructions. We are going to write computer programs to solve problems. Problem Solving Problem: gap between the existing and the desired or proposed situation where problem solving will try to fill this gap. Problem solving: The process of transforming the description of a problem into the solution by using our knowledge of the problem-solving strategies, techniques and tools. Problem solving techniques • Analyze the problem: Carefully understand the problem • Outline the problem requirements • Does program require user interaction? • Does program manipulate data? • What is the output? • Design steps (algorithm) to solve the problem

Question sources

Computer Science MCQ practice collection · Question 252

Question 115 | AM0253

What is the role of comments in programming?

A. To execute specific sections of code

B. To define variables and constants

C. To provide documentation and explanations within the code

D. To display output to the user

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: To provide documentation and explanations within the code

Comments are descriptive text explaining aspects of a program, ignored by the compiler and intended for human readers (p.5). Thus they document/explain code; they do not execute, declare variables by themselves or display text at runtime. Module support: PDF p. 5.

Why each choice is right or wrong

A - Incorrect: "To execute specific sections of code" does not match the rule or result derived here. Comments are descriptive text explaining aspects of a program, ignored by the compiler and intended for human readers (p.5). Thus they document/explain code; they do not execute, declare variables by themselves or display text at runtime.

B - Incorrect: "To define variables and constants" does not match the rule or result derived here. Comments are descriptive text explaining aspects of a program, ignored by the compiler and intended for human readers (p.5). Thus they document/explain code; they do not execute, declare variables by themselves or display text at runtime.

C - Correct: "To provide documentation and explanations within the code" matches the derived result. Comments are descriptive text explaining aspects of a program, ignored by the compiler and intended for human readers (p.5). Thus they document/explain code; they do not execute, declare variables by themselves or display text at runtime.

D - Incorrect: "To display output to the user" does not match the rule or result derived here. Comments are descriptive text explaining aspects of a program, ignored by the compiler and intended for human readers (p.5). Thus they document/explain code; they do not execute, declare variables by themselves or display text at runtime.

Primary note proof

Study-note section CP05

> Comments use // to the line end or /* ... */ for a block. They explain intent and assumptions for maintenance; ordinary comments do not execute or automatically improve speed, reliability or portability. main is the conventional entry function name, not a C++ keyword.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 18-20

> 4. Comments A comment is a piece of descriptive text which explains some aspect of a program. Program comments are totally ignored by the compiler and are only intended for human readers.

Question sources

Computer Science MCQ practice collection · Question 253

Question 116 | AM0254

What is the purpose of version control in programming?

A. To write programs

B. To execute programs

C. To manage and track changes in code over time

D. To optimize program performance

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Manage and track code changes over time.

CP32 says version control records changes and supports comparison and rollback. That is the management of code history described by C. A version-control system may support a development workflow, but it does not itself define the process of writing code, executing a program or optimizing its runtime performance.

Why each choice is right or wrong

A - Incorrect: Developers write programs; version control records and manages the resulting revisions rather than creating program logic.

B - Incorrect: Execution is performed by the program's runtime or machine, not by revision-history management.

C - Correct: Tracking revisions, comparing changes and rolling back are the version-control purposes explicitly stated in CP32.

D - Incorrect: Runtime performance optimization changes program efficiency; version control manages the history of such changes.

Primary note proof

Study-note section CP32

> Test ordinary, boundary and invalid inputs and trace the first point where actual state differs from expected state. Exceptions transfer control to a matching handler for exceptional conditions; they do not make arbitrary undefined behavior recoverable. Refactoring improves internal structure while preserving observable behavior. Version control records changes, supports comparison and rollback; documentation explains use, design and maintenance assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 2-18

1. Introduction to programming Computer programming is the process of writing, testing, debugging/troubleshooting, and maintaining the source code of computer programs. Programs are the instructions that tells the computer what to do .so, writing computer programs means writing instructions, that will make the computer follow and run a program based on those instructions. We are going to write computer programs to solve problems. Problem Solving Problem: gap between the existing and the desired or proposed situation where problem solving will try to fill this gap. Problem solving: The process of transforming the description of a problem into the solution by using our knowledge of the problem-solving strategies, techniques and tools. Problem solving techniques • Analyze the problem: Carefully understand the problem • Outline the problem requirements • Does program require user interaction? • Does program manipulate data? • What is the output? • Design steps (algorithm) to solve the problem

Question sources

Computer Science MCQ practice collection · Question 254

Question 117 | AM0255

What is the difference between a parameter and an argument in programming?

A. Parameters are used in functions, while arguments are used in loops.

B. Parameters are passed to functions, while arguments are used to define functions.

C. Parameters are variables used to define a function, while arguments are the values passed to the function when it is called.

D. There is no difference; the terms are used interchangeably.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Parameters are variables used to define a function, while arguments are the values passed to the function when it is called.

The definition syntax has a Parameter List (p.15), whereas the call Addition(x,y) supplies values; the passing-by-value discussion calls the passed actual value an argument (p.16). This supports C. A assigns arguments to loops, B reverses definition/call roles, and D ignores the distinction illustrated. Module support: PDF p. 15, PDF p. 16.

Why each choice is right or wrong

A - Incorrect: "Parameters are used in functions, while arguments are used in loops." does not match the rule or result derived here. The definition syntax has a Parameter List (p.15), whereas the call Addition(x,y) supplies values; the passing-by-value discussion calls the passed actual value an argument (p.16). This supports C. A assigns arguments to loops, B reverses definition/call roles, and D ignores the distinction illustrated.

B - Incorrect: "Parameters are passed to functions, while arguments are used to define functions." does not match the rule or result derived here. The definition syntax has a Parameter List (p.15), whereas the call Addition(x,y) supplies values; the passing-by-value discussion calls the passed actual value an argument (p.16). This supports C. A assigns arguments to loops, B reverses definition/call roles, and D ignores the distinction illustrated.

C - Correct: "Parameters are variables used to define a function, while arguments are the values passed to
the function when it is called." matches the derived result. The definition syntax has a Parameter List (p.15), whereas the call Addition(x,y) supplies values; the passing-by-value discussion calls the passed actual value an argument (p.16). This supports C. A assigns arguments to loops, B reverses definition/call roles, and D ignores the distinction illustrated.

D - Incorrect: "There is no difference; the terms are used interchangeably." does not match the rule or result derived here. The definition syntax has a Parameter List (p.15), whereas the call Addition(x,y) supplies values; the passing-by-value discussion calls the passed actual value an argument (p.16). This supports C. A assigns arguments to loops, B reverses definition/call roles, and D ignores the distinction illustrated.

Primary note proof

Study-note section CP17

> A function groups a named task. C++ definition: int add(int a,int b){return a+b;}. Call it with add(x,y);, without parameter type declarations at the call site. Parameters are names in the definition; arguments are supplied expressions or values. A void function produces no return value but can have side effects and return; to exit early.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 29-46

> • User-defined - Function that been created by the user. - These functions need to be declared and defined by the user. Declaration of function Function defines using the following syntax: Function_type function_name (Parameter List) { ...function body } Example: void Addition(int num1,int num2) #include<iostream> using namespace std; void Addition(int num1,int num2) { int sum=num1+num2; cout<<"The sum is:"<<sum; } int main() { int x=10,y=20; Addition(x,y);

Computer Programming module | PDF page 16 | lines 13-20

> Passing by Value In call by value, the actual value that is passed as argument is not changed after performing some operation on it. it creates a copy of that variable into the stack section in memory. When the value is changed, it changes the value of that copy, the actual value remains the same Example: Passing by Reference: It allows a function to modify a variable without having to create a copy of it. We have to declare reference variables. The memory location of the passed variable and parameter is the same and therefore, any change to the parameter reflects in the variable as well.

Question sources

Computer Science MCQ practice collection · Question 255

Question 118 | AM0256

What is the purpose of logical operators in programming?

A. To perform arithmetic calculations

B. To control the flow of program execution

C. To combine or manipulate boolean values

D. To create user interfaces

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: To combine or manipulate boolean values

The module lists !, && and || as logical operators producing 1/0 truth outcomes (p.7). They manipulate and combine truth values. Arithmetic operators perform numerical calculations; UI construction and flow control are broader/different roles. Module support: PDF p. 7.

Why each choice is right or wrong

A - Incorrect: "To perform arithmetic calculations" does not match the rule or result derived here. The module lists !, && and || as logical operators producing 1/0 truth outcomes (p.7). They manipulate and combine truth values. Arithmetic operators perform numerical calculations; UI construction and flow control are broader/different roles.

B - Incorrect: "To control the flow of program execution" does not match the rule or result derived here. The module lists !, && and || as logical operators producing 1/0 truth outcomes (p.7). They manipulate and combine truth values. Arithmetic operators perform numerical calculations; UI construction and flow control are broader/different roles.

C - Correct: "To combine or manipulate boolean values" matches the derived result. The module lists !, && and || as logical operators producing 1/0 truth outcomes (p.7). They manipulate and combine truth values. Arithmetic operators perform numerical calculations; UI construction and flow control are broader/different roles.

D - Incorrect: "To create user interfaces" does not match the rule or result derived here. The module lists !, && and || as logical operators producing 1/0 truth outcomes (p.7). They manipulate and combine truth values. Arithmetic operators perform numerical calculations; UI construction and flow control are broader/different roles.

Primary note proof

Study-note section CP10

> Relational and equality operators test values; && and || combine truth values; ! negates. In C++, zero converts to false and a nonzero number to true. Streams display bool as 0 or 1 unless boolalpha is set. With a=3,b=5, cout<<(a==b) prints 0.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science MCQ practice collection · Question 256

Question 119 | AM0257

What is the purpose of modular programming?

A. To create user interfaces

B. To write efficient code

C. To divide a program into separate, reusable modules

D. To optimize program performance

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: To divide a program into separate, reusable modules

The module says functions structure programs modularly, and dividing a complex problem into smaller chunks improves understanding and reuse (p.14). That directly supports separate reusable modules. Efficiency may benefit indirectly but is not the stated modular decomposition purpose. Module support: PDF p. 14.

Why each choice is right or wrong

A - Incorrect: "To create user interfaces" does not match the rule or result derived here. The module says functions structure programs modularly, and dividing a complex problem into smaller chunks improves understanding and reuse (p.14). That directly supports separate reusable modules. Efficiency may benefit indirectly but is not the stated modular decomposition purpose.

B - Incorrect: "To write efficient code" does not match the rule or result derived here. The module says functions structure programs modularly, and dividing a complex problem into smaller chunks improves understanding and reuse (p.14). That directly supports separate reusable modules. Efficiency may benefit indirectly but is not the stated modular decomposition purpose.

C - Correct: "To divide a program into separate, reusable modules" matches the derived result. The module says functions structure programs modularly, and dividing a complex problem into smaller chunks improves understanding and reuse (p.14). That directly supports separate reusable modules. Efficiency may benefit indirectly but is not the stated modular decomposition purpose.

D - Incorrect: "To optimize program performance" does not match the rule or result derived here. The module says functions structure programs modularly, and dividing a complex problem into smaller chunks improves understanding and reuse (p.14). That directly supports separate reusable modules. Efficiency may benefit indirectly but is not the stated modular decomposition purpose.

Primary note proof

Study-note section CP17

> A method is a function associated with a class or object; methods can return values. Modules group related responsibilities behind interfaces. Reuse reduces duplication; independent functions are easier to understand and test. A library provides reusable components; a framework also supplies an organizing structure and often calls application callbacks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 14 | lines 21-33

> Function A function is a group of statements that structure our programs in a more modular way and executed when it is called from some point of the program. It is a block of code that performs a specific task. It is important to Dividing a complex problem into smaller chunks makes our program easy to understand and reusable. When we need function? - When you need to repeat the same process over and over in a program. - The function can be called many times but appears in the code once Function - Categories C++ functions can be divided into 2 categories: • pre-defined (standard function) / library function - Which is the definitions have been written and it is ready to be used. - User needs to include pre-defined header file (i.e., math.h, time.h,cmath) To use it: FunctionName (value) e.g., sqrt (4), floor (4.3)

Question sources

Computer Science MCQ practice collection · Question 257

Question 120 | AM0258

What is the role of data structures in programming?

A. To store and organize data efficiently

B. To control the flow of program execution

C. To define user interfaces

D. To perform input/output operations

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: To store and organize data efficiently

The module organizes homogeneous indexed values as arrays (p.19) and grouped fields as structures (pp.34-35). These are concrete ways of storing and organizing data, matching A. Flow control, interface definition and input/output are separate program operations, not the storage organization role. Module support: PDF p. 19, PDF p. 35.

Why each choice is right or wrong

A - Correct: "To store and organize data efficiently" matches the derived result. The module organizes homogeneous indexed values as arrays (p.19) and grouped fields as structures (pp.34-35). These are concrete ways of storing and organizing data, matching A. Flow control, interface definition and input/output are separate program operations, not the storage organization role.

B - Incorrect: "To control the flow of program execution" does not match the rule or result derived here. The module organizes homogeneous indexed values as arrays (p.19) and grouped fields as structures (pp.34-35). These are concrete ways of storing and organizing data, matching A. Flow control, interface definition and input/output are separate program operations, not the storage organization role.

C - Incorrect: "To define user interfaces" does not match the rule or result derived here. The module organizes homogeneous indexed values as arrays (p.19) and grouped fields as structures (pp.34-35). These are concrete ways of storing and organizing data, matching A. Flow control, interface definition and input/output are separate program operations, not the storage organization role.

D - Incorrect: "To perform input/output operations" does not match the rule or result derived here. The module organizes homogeneous indexed values as arrays (p.19) and grouped fields as structures (pp.34-35). These are concrete ways of storing and organizing data, matching A. Flow control, interface definition and input/output are separate program operations, not the storage organization role.

Primary note proof

Study-note section CP22

A built-in C++ array has a fixed number of same-type elements in contiguous storage: int a[4]={10,20,30,40};. Valid indices are 0 through 3. a[4] is outside the array; accessing it has undefined behavior. Java checks bounds and throws ArrayIndexOutOfBoundsException. Do not predict a C++ out-of-range value from nearby memory.

Study-note section CP27

A struct groups named members, possibly of different types. struct House{double price; int rooms;}; defines a type; House h{150000,3}; creates an object. h.rooms uses a dot. House* p=&h; p->rooms is equivalent to (*p).rooms. The pointer must refer to a live object before member access. A type definition alone does not create an ordinary instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 19 | lines 4-29

Arrays - an array is a collection of fixed-size number of elements of the same type stored sequentially in memory. - An individual element of an array is identified by its own unique index (or subscript). - The number of elements must be an integer. - The size of the array is referred to as its dimension - Array in C++ is Zero indexed. Declaring of one-dimensional Array Syntax: elementstype arrayname [size]; → You can use constants such as macro definitions or const ints as the sizes of arrays. Example: int num[5]; string fruit[3]; #define numstudents 100 or const int numstudents=100; double Score[numstudents] ; double Score[numstudents] ; Initialization of an Array - Arrays can be initialized at declaration time in two different ways: Syntax: elementstype arrayname[size] = {list each value separated by comma}; E.g int num[5]={1,2,3,4,5}; elementstype arrayname[ ] = { list with any number of values }; E.g int num[]={1,2,3,4,5}; string fruit[]={“Apple”,“orange”, “Mango”}; Accessing Array Elements - Each element could be accessed using an index into the array - When accessing an array, the index given must be a positive integer from 0 to n-1, where n is the dimension of the array.

Computer Programming module | PDF page 35 | lines 13-17

Accessing members of a structure variable A structure member can be accessed using dot operator’.’ notation syntax: structvariable.member; eg. st.id=4;

Question sources

Computer Science MCQ practice collection · Question 258

Question 121 | AM0259

What is the purpose of exception handling in programming?

A. To perform input/output operations

B. To optimize program performance

C. To handle and recover from runtime errors

D. To define variables and constants

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Handle exceptional runtime conditions and recover where appropriate.

CP32 describes exceptions as transferring control to a matching handler for exceptional conditions. This supports handling and possible recovery from runtime errors. It does not promise that every runtime fault is recoverable: arbitrary undefined behavior is specifically excluded. Input/output, optimization and declarations are not exception handling's defining purpose.

Why each choice is right or wrong

A - Incorrect: Input/output may raise exceptional conditions, but performing I/O is not the defining purpose of exception handling.

B - Incorrect: Exception handling routes exceptional control flow rather than being a mechanism for performance optimization.

C - Correct: Matching handlers support handling and possible recovery from runtime exceptions, subject to the note's stated limits.

D - Incorrect: Variables and constants are introduced by declarations, not by exception-handling control flow.

Primary note proof

Study-note section CP32

> Test ordinary, boundary and invalid inputs and trace the first point where actual state differs from expected state. Exceptions transfer control to a matching handler for exceptional conditions; they do not make arbitrary undefined behavior recoverable. Refactoring improves internal structure while preserving observable behavior. Version control records changes, supports comparison and rollback; documentation explains use, design and maintenance assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 9 | lines 2-8

> Control Statements Control statements or flow control determines what is executed during a run and what is not, therefore affecting the overall outcome of the program. C++ provides different forms of statements for different purposes. Declaration statements are used for defining variables. Assignment-like statements are used for simple, algebraic computations. Branching statements are used for specifying alternate paths of execution, depending on the outcome of a logical condition. Loop statements are used for specifying computations, which need to be repeated until a certain logical condition is satisfied

Question sources

Computer Science MCQ practice collection · Question 259

Question 122 | AM0260

Which statement accurately distinguishes static typing from dynamic typing? Consider type-checking time rather than mandatory annotations or programming paradigms.

A. Static typing is more efficient than dynamic typing.

B. Static typing checks types before execution, possibly using inferred types; dynamic typing checks operations using runtime values.

C. Static typing is used in object-oriented programming, while dynamic typing is used in procedural programming.

D. There is no difference; the terms are used interchangeably.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: B. Static typing checks types before execution, possibly using inferred types; dynamic typing checks operations using runtime values.

Static typing checks types before execution, and the static type may be inferred rather than explicitly written. Dynamic typing checks operations using values at runtime. The difference is therefore about the typing/checking model, not a guarantee of execution speed or a choice between object-oriented and procedural programming. B states this distinction; static typing does not require explicitly written type declarations.

Why each choice is right or wrong

A - Incorrect: Typing discipline alone does not guarantee a universal efficiency ranking.

B - Correct: This correctly distinguishes before-execution checking from runtime checking and allows inference.

C - Incorrect: Object-oriented versus procedural organization does not determine typing discipline.

D - Incorrect: The two checking models have distinct meanings.

Primary note proof

Study-note section CP33

> Static typing checks types before execution; type inference can still infer a static type, as with C++ auto. Dynamic typing checks operations using runtime values; it does not mean variables must have explicit static declarations. Static typing does not always require explicit type annotations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 24-33

5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Question sources

Computer Science MCQ practice collection · Question 260

Question 123 | AM0261

What is the purpose of libraries and frameworks in programming?

A. To perform input/output operations

B. To optimize program performance

C. To provide reusable code and functionality

D. To create user interfaces

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: To provide reusable code and functionality

Library functions are described as predefined definitions ready to use, with headers required before calling them (p.14). This supplies reusable functionality rather than forcing users to implement it again. Input/output and UI are possible particular facilities, not the broad purpose; optimization is not the definition. Module support: PDF p. 14, PDF p. 15.

Why each choice is right or wrong

A - Incorrect: "To perform input/output operations" does not match the rule or result derived here. Library functions are described as predefined definitions ready to use, with headers required before calling them (p.14). This supplies reusable functionality rather than forcing users to implement it again. Input/output and UI are possible particular facilities, not the broad purpose; optimization is not the definition.

B - Incorrect: "To optimize program performance" does not match the rule or result derived here. Library functions are described as predefined definitions ready to use, with headers required before calling them (p.14). This supplies reusable functionality rather than forcing users to implement it again. Input/output and UI are possible particular facilities, not the broad purpose; optimization is not the definition.

C - Correct: "To provide reusable code and functionality" matches the derived result. Library functions are described as predefined definitions ready to use, with headers required before calling them (p.14). This supplies reusable functionality rather than forcing users to implement it

again. Input/output and UI are possible particular facilities, not the broad purpose; optimization is not the definition.

D - Incorrect: "To create user interfaces" does not match the rule or result derived here. Library functions are described as predefined definitions ready to use, with headers required before calling them (p.14). This supplies reusable functionality rather than forcing users to implement it again. Input/output and UI are possible particular facilities, not the broad purpose; optimization is not the definition.

Primary note proof

Study-note section CP17

A method is a function associated with a class or object; methods can return values. Modules group related responsibilities behind interfaces. Reuse reduces duplication; independent functions are easier to understand and test. A library provides reusable components; a framework also supplies an organizing structure and often calls application callbacks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 14 | lines 21-33

Function A function is a group of statements that structure our programs in a more modular way and executed when it is called from some point of the program. It is a block of code that performs a specific task. It is important to Dividing a complex problem into smaller chunks makes our program easy to understand and reusable. When we need function? - When you need to repeat the same process over and over in a program. - The function can be called many times but appears in the code once Function - Categories C++ functions can be divided into 2 categories: • pre-defined (standard function) / library function - Which is the definitions have been written and it is ready to be used. - User needs to include pre-defined header file (i.e., math.h, time.h,cmath) To use it: FunctionName (value) e.g., sqrt (4), floor (4.3)

Computer Programming module | PDF page 15 | lines 29-46

• User-defined - Function that been created by the user. - These functions need to be declared and defined by the user. Declaration of function Function defines using the following syntax: Function_type function_name (Parameter List) { ...function body } Example: void Addition(int num1,int num2) #include<iostream> using namespace std; void Addition(int num1,int num2) { int sum=num1+num2; cout<<"The sum is:"<<sum; } int main() { int x=10,y=20; Addition(x,y);

Question sources

Computer Science MCQ practice collection · Question 261

Question 124 | AM0262

What is the purpose of code refactoring in programming?

A. To write efficient code

B. To optimize program performance

C. To improve the readability and maintainability of code

D. To handle and recover from runtime errors

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: Improve code readability and maintainability while preserving behavior.

CP32 defines refactoring as improving internal structure while preserving observable behavior. Better readability and maintainability are the offered purpose that matches this definition. Performance tuning can accompany other changes, but is not the defining aim; runtime-error recovery belongs to exception handling rather than refactoring.

Why each choice is right or wrong

A - Incorrect: Writing efficient code is broader than refactoring and does not specify preserving existing observable behavior.

B - Incorrect: Performance optimization has a different primary aim; refactoring concerns structure and maintainability.

C - Correct: Readability and maintainability improve through internal restructuring while observable program behavior is preserved.

D - Incorrect: Handling exceptional runtime conditions belongs to exception handling, not to refactoring's structural purpose.

Primary note proof

Study-note section CP32

> Test ordinary, boundary and invalid inputs and trace the first point where actual state differs from expected state. Exceptions transfer control to a matching handler for exceptional conditions; they do not make arbitrary undefined behavior recoverable. Refactoring improves internal structure while preserving observable behavior. Version control records changes, supports comparison and rollback; documentation explains use, design and maintenance assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 2-18

> 1. Introduction to programming Computer programming is the process of writing, testing, debugging/troubleshooting, and maintaining the source code of computer programs. Programs

are the instructions that tells the computer what to do .so, writing computer programs means writing instructions, that will make the computer follow and run a program based on those instructions. We are going to write computer programs to solve problems. Problem Solving Problem: gap between the existing and the desired or proposed situation where problem solving will try to fill this gap. Problem solving: The process of transforming the description of a problem into the solution by using our knowledge of the problem-solving strategies, techniques and tools. Problem solving techniques • Analyze the problem: Carefully understand the problem • Outline the problem requirements • Does program require user interaction? • Does program manipulate data? • What is the output? • Design steps (algorithm) to solve the problem

Computer Programming module | PDF page 5 | lines 18-20

4. Comments A comment is a piece of descriptive text which explains some aspect of a program. Program comments are totally ignored by the compiler and are only intended for human readers.

Question sources

Computer Science MCQ practice collection · Question 262

Question 125 | AM0263

What is the role of algorithms in programming?

A. To store and organize data efficiently

B. To perform input/output operations

C. To define user interfaces

D. To provide step-by-step instructions for solving problems

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: To provide step-by-step instructions for solving problems

The module defines an algorithm as ordered unambiguous executable steps defining a terminating process (p.2). Therefore it supplies step-by-step problem-solving instructions. Storage organization, input/output and UI design are possible concerns of a program rather than this general definition. Module support: PDF p. 2.

Why each choice is right or wrong

A - Incorrect: "To store and organize data efficiently" does not match the rule or result derived here. The module defines an algorithm as ordered unambiguous executable steps defining a terminating process (p.2). Therefore it supplies step-by-step problem-solving instructions. Storage organization, input/output and UI design are possible concerns of a program rather than this general definition.

B - Incorrect: "To perform input/output operations" does not match the rule or result derived here. The module defines an algorithm as ordered unambiguous executable steps defining a terminating process (p.2). Therefore it supplies step-by-step problem-solving instructions. Storage organization, input/output and UI design are possible concerns of a program rather than this general definition.

C - Incorrect: "To define user interfaces" does not match the rule or result derived here. The module defines an algorithm as ordered unambiguous executable steps defining a terminating process (p.2). Therefore it supplies step-by-step problem-solving instructions. Storage organization, input/output and UI design are possible concerns of a program rather than this general definition.

D - Correct: "To provide step-by-step instructions for solving problems" matches the derived result. The module defines an algorithm as ordered unambiguous executable steps defining a terminating process (p.2). Therefore it supplies step-by-step problem-solving instructions. Storage organization, input/output and UI design are possible concerns of a program rather than this general definition.

Primary note proof

Study-note section CP02

> An algorithm gives precise, executable steps and terminates for every legal input. Definiteness means each step has an unambiguous meaning; effectiveness means a step can actually be performed. Inputs may be absent, but a useful algorithm specifies a result. Correctness means the result meets the specification. Efficiency is desirable and measurable, but an inefficient terminating algorithm is still an algorithm. Efficiency is also a desirable property of an algorithm.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 2 | lines 19-26

> Algorithm is an ordered set of unambiguous, executable steps, defining a terminating process. Important Properties of Algorithms - Correct: always returns the desired output for all legal instances of the problem. - Unambiguous: Each step must be clearly defined, having only one interpretation. - Precise: simple but efficient - Efficient: can be measured in terms of Time and Space - Finiteness: the execution of the algorithm must terminate in a finite amount of time for each possible set of input data/must have an end.

Question sources

Computer Science MCQ practice collection · Question 263

Question 126 | AM0264

What is the purpose of documentation in programming?

A. To execute programs

B. To optimize program performance

C. To provide explanations and instructions for using and maintaining the code

D. To create user interfaces

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: To provide explanations and instructions for using and maintaining the code

The module defines comments as descriptive text explaining a program for human readers (p.5). Documentation serves this explanatory role; instructions for use and maintenance are a reasonable application of that role. Module support: PDF p. 5.

Why each choice is right or wrong

A - Incorrect: Execution follows program instructions, whereas explanatory comments are ignored by the compiler.

B - Incorrect: The cited purpose is human understanding, not performance optimization.

C - Correct: Explanations for readers support using and maintaining code.

D - Incorrect: Creating interfaces is not the descriptive role defined on p.5.

Primary note proof

Study-note section CP05

> Comments use // to the line end or /* ... */ for a block. They explain intent and assumptions for maintenance; ordinary comments do not execute or automatically improve speed, reliability or portability. main is the conventional entry function name, not a C++ keyword.

Study-note section CP32

> Test ordinary, boundary and invalid inputs and trace the first point where actual state differs from expected state. Exceptions transfer control to a matching handler for exceptional conditions; they do not make arbitrary undefined behavior recoverable. Refactoring improves internal structure while preserving observable behavior. Version control records changes, supports comparison and rollback; documentation explains use, design and maintenance assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 18-20

4. Comments A comment is a piece of descriptive text which explains some aspect of a program. Program comments are totally ignored by the compiler and are only intended for human readers.

Question sources

Computer Science MCQ practice collection · Question 264

Question 127 | AM0265

What does the cout object do in C++?

A. It reads input from the user.

B. It writes output to the console.

C. It performs mathematical calculations.

D. It declares variables.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: It writes output to the console.

Page 6 explicitly calls cout the standard output stream and says it displays information for the user. Module support: PDF p. 6.

Why each choice is right or wrong

A - Incorrect: Input is assigned to cin on p.6.

B - Correct: This is the stated output role.

C - Incorrect: cout displays expressions; it is not the calculation operator.

D - Incorrect: Declarations use a data type and name (p.5), not cout.

Primary note proof

Study-note section CP28

> Include <iostream> and qualify std::cin/std::cout or introduce the names explicitly. cout is the standard output stream; << inserts values. cin is standard input; >> extracts formatted values. std::endl inserts a newline and flushes. C++ names are case sensitive: Cout, Int and std: :cout are not equivalent valid spellings of cout, int and std::cout.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 6 | lines 4-15

> 6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Computer Science MCQ practice collection · Question 265

Question 128 | AM0266

Without any user-defined type aliases, which option declares an ordinary C++ int variable named variableName?

A. variableName = value;

B. value = variableName;

C. int variableName;

D. variableName var;

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: C. int variableName;

An ordinary C++ variable declaration places a type specifier before the variable name. Option C uses int followed by variableName and a semicolon, so it introduces the requested integer variable. The assignment expressions in A and B do not introduce a type. D would require variableName to be a type, contrary to the stated requested variable name and absence of aliases.

Why each choice is right or wrong

A - Incorrect: An assignment does not declare the variable's type.

B - Incorrect: Reversing an assignment still does not introduce a declaration.

C - Correct: int variableName; has the required type-then-name form.

D - Incorrect: variableName is not a type under the stated assumptions.

Primary note proof

Study-note section CP06

> Write type name; or type name = value; for ordinary variables: int age = 20;. Declaration introduces a name and type; initialization gives its initial value; later assignment replaces a value. a=b copies a scalar value, so later changes to b do not change a. A pointer or Java object reference instead copies an address/reference.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 24-33

> 5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Question sources

Computer Science MCQ practice collection · Question 266

Question 129 | AM0267

What is the purpose of the for loop in C++?

A. To perform conditional branching.

B. To read input from the user.

C. To execute a block of code repeatedly.

D. To declare variables.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: To execute a block of code repeatedly.

The for syntax on p.12 checks a condition, executes the body, performs the increase, then returns to the condition. This gives repeated execution. Module support: PDF p. 12.

Why each choice is right or wrong

A - Incorrect: Branch selection is not the loop purpose.

B - Incorrect: Input uses cin.

C - Correct: The body repeats while the condition holds.

D - Incorrect: An initializer can declare a counter, but declaration is not the loop purpose.

Primary note proof

Study-note section CP14

> for(initialization; condition; update) executes initialization once, then condition → body → update → condition. A missing condition in for(;;) acts as true. A for loop is convenient for a known count but can also be condition controlled. To print indices 0 through n-1, use i=0; i<n; ++i. Starting i=0 with i<5 and i+=2 prints 0 2 4.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 19-28

> The for loop. Its format is: for (initialization; condition; increase) statement; 1, initialization is executed. Generally, it is a initial value setting for a counter variable. This is executed only once 2, condition is checked, if it is true the loop continues, otherwise the loop finishes and statement is skipped. 3, statement is executed. As usual, it can be either a single instruction or a block of instructions enclosed within curly brackets { }. 4, finally, whatever is specified in the increase field is executed and the loop gets back to step 2.

Question sources

Computer Science MCQ practice collection · Question 267

Question 130 | AM0268

What is the output of the following code snippet?

```
int x = 5;
cout << x++ << endl;
cout << x << endl;
```

A. 5, 6

B. 5, 5

C. 6, 6

D. 6, 5

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer A: 5, 6

With x initially 5, postfix x++ yields its old value and then increases x to 6; the next statement displays 6. The module operator example on p.8 contrasts postfix b++ and prefix ++a; its previously inspected output confirms this distinction. The two cout statements thus print 5 then 6 on separate lines. Module support: PDF p. 7, PDF p. 8, PDF p. 6.

Why each choice is right or wrong

A - Correct: Old value followed by updated value.

B - Incorrect: Fails to include the increment in the second output.

C - Incorrect: Treats postfix as prefix in the first output.

D - Incorrect: Reverses both the initial output and final state.

Primary note proof

Study-note section CP09

> Prefix ++x changes x and yields the new value; postfix x++ yields the previous value and also increments x. Separate the value used in the expression from the stored value after it. Starting a=8,b=5, a++ + --b uses 8+4, gives 12, and leaves a=9,b=4. Starting x=1,y=10,z=-3, (x++*y--)+ ++z gives 8 and leaves x=2,y=9,z=-2.

Study-note section CP28

> std::endl inserts a newline and flushes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y,

Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 8 | lines 12-23

{int a=10,b=3,c,d,e,f; c=a%b++; cout<<"c="<<c<<endl; b+=2; cout<<"b="<<b<<endl; d=++a+b; cout<<"d="<<d<<endl; e=a>b?a:b; e++; cout<<"e="<<e++<<endl; f=a+b+c+d+e; cout<<"f="<<f;

Computer Programming module | PDF page 6 | lines 4-15

6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Computer Science MCQ practice collection · Question 268

Question 131 | AM0269

What is the purpose of the switch statement in C++?

A. To perform mathematical calculations.

B. To read input from the user.

C. To execute a block of code repeatedly.

D. To perform different actions based on different conditions.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: To perform different actions based on different conditions.

Page 10 defines switch as choosing among alternatives based on an expression value; p.11 describes matching that value to case labels. D is the matching description, with conditions understood as case matches. Module support: PDF p. 10.

Why each choice is right or wrong

A - Incorrect: Arithmetic is not switch selection.

B - Incorrect: Input is performed by cin.

C - Incorrect: Repetition belongs to loops.

D - Correct: Selecting actions by expression values matches the definition.

Primary note proof

Study-note section CP13

> A traditional C++ switch selects using an integral or enumeration value. case labels are constant expressions. Execution starts at the matching label, then continues through subsequent statements until a break, return or the end. default handles an unmatched value but need not be the last physical label.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 10 | lines 15-26

> switch Statement The switch statement provides a way of choosing between a set of alternatives, based on the value of an expression. The general form of the switch statement is: switch (expression) { caseconstant1: statements; ... caseconstantn: statements; default: statements; }

Question sources

Computer Science MCQ practice collection · Question 269

Question 132 | AM0270

With no user-defined return-type aliases, which C++ function definition can be called and then reach its closing brace normally without omitting a required return value? Assume this is an ordinary function other than main.

A. function myFunction() { }

B. void myFunction() { }

C. int myFunction() { }

D. myFunction() { }

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: B. void myFunction() { }

The void definition needs no result value and can safely finish at the end of its empty body. The int definition can be accepted as a definition, but reaching its end without returning a value has undefined behavior. The other forms omit a known return type. Specifying normal completion of an ordinary call distinguishes a safe complete definition from merely syntactically accepted but unsafe fall-through.

Why each choice is right or wrong

A - Incorrect: function is not a declared return type or C++ definition keyword here.

B - Correct: A void function can finish its empty body without returning a value.

C - Incorrect: Reaching the end of this int function omits a required return value.

D - Incorrect: An ordinary C++ function requires an explicit return type.

Primary note proof

Study-note section CP17

> A function groups a named task. C++ definition: int add(int a,int b){return a+b;}. Call it with add(x,y);, without parameter type declarations at the call site. Parameters are names in the definition; arguments are supplied expressions or values. A void function produces no return value but can have side effects and return; to exit early.

Study-note section CP35

> A void function definition can be complete without returning a value; int myFunction(){} can compile, but reaching its end is undefined behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 29-46

• User-defined - Function that been created by the user. - These functions need to be declared and defined by the user. Declaration of function Function defines using the following syntax: Function_type function_name (Parameter List) { ...function body } Example: void Addition(int num1,int num2) #include<iostream> using namespace std; void Addition(int num1,int num2) { int sum=num1+num2; cout<<"The sum is:"<<sum; } int main() { int x=10,y=20; Addition(x,y);

Question sources

Computer Science MCQ practice collection · Question 270

Question 133 | AM0271

What is the purpose of the sizeof operator in C++?

A. To determine the memory size of a variable or data type.

B. To perform mathematical calculations.

C. To compare two variables.

D. To check the equality of two values.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: To determine the memory size of a variable or data type.

Page 29 ties pointer scaling to the size in bytes of the pointed object, and its example labels sizeof(x) as size of int. This supports sizeof as memory-size measurement, rather than arithmetic or comparison. Module support: PDF p. 29.

Why each choice is right or wrong

A - Correct: Memory size is the role shown by size of int and sizeof(x).

B - Incorrect: The example measures storage rather than calculating an arithmetic expression.

C - Incorrect: Comparisons use relational operators, not sizeof.

D - Incorrect: Equality uses == (p.7), not sizeof.

Primary note proof

Study-note section CP07

Single quotes denote a character literal, such as 'A'; double quotes denote a string literal, such as "A". A string includes more storage than just its visible characters when a terminator is required. sizeof reports size in bytes, with sizeof(char)==1; the byte width and other sizes are implementation dependent. sizeof(array) gives total array storage before conversion to a pointer.

Study-note section CP10

Relational and equality operators test values; && and || combine truth values; ! negates.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 29 | lines 6-17

> Note that ptr + 1 does not return the memory address after ptr, but the memory address of the next object of the type that ptr points to. If ptr points to an integer (assuming 4 bytes), ptr + 3 means 3 integers (12 bytes) after ptr. If ptr points to a char, which is always 1 byte, ptr + 3 means 3 chars (3 bytes) after ptr. When calculating the result of a pointer arithmetic expression, the compiler always multiplies the integer operand by the size of the object being pointed to. This is called scaling. Example: #include <iostream> using namespace std; int main() { int x=7; int* ptr=&x; cout<<"size of int:"<<sizeof(x)<<endl;

Variant 1 | Alternative version 1

What is the purpose of the sizeof operator in C++?

A. To determine the size of a variable in bytes.

B. To perform mathematical calculations.

C. To compare two variables.

D. To check the equality of two values.

Answer A: To determine the size of a variable in bytes.

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Correct: The pointer-scaling discussion uses bytes and the example labels sizeof(x) as size of int (p.29), supporting size of a variable in bytes.

B - Incorrect: sizeof measures storage size in the example, not arithmetic.

C - Incorrect: Comparing values uses relational operators (p.7).

D - Incorrect: Equality uses ==, not sizeof (p.7).

Primary note proof

Study-note section CP07

> Single quotes denote a character literal, such as 'A'; double quotes denote a string literal, such as "A". A string includes more storage than just its visible characters when a terminator is required. sizeof reports size in bytes, with sizeof(char)==1; the byte width and other sizes are implementation dependent. sizeof(array) gives total array storage before conversion to a pointer.

Study-note section CP10

> Relational and equality operators test values; && and || combine truth values; ! negates.

Question sources

Computer Science MCQ practice collection · Question 271

Computer Science MCQ practice collection · Question 278

Question 134 | AM0272

What does break do when executed inside a C++ loop or switch?

A. To terminate the program.

B. To exit the nearest enclosing loop or switch.

C. To repeat the loop from the beginning.

D. To execute a specific case in a switch statement.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: B. To exit the nearest enclosing loop or switch.

break exits the nearest enclosing loop or switch and resumes control after that construct. It does not terminate the entire program merely because it leaves the construct. Skipping only the remainder of the current iteration is the role of continue, while selecting a case is the switch expression's role. Option B states the exact enclosing-construct exit behavior rather than confusing it with iteration skipping.

Why each choice is right or wrong

A - Incorrect: Leaving a loop or switch does not itself terminate the program.

B - Correct: break exits the nearest enclosing loop or switch.

C - Incorrect: break exits rather than restarting the loop.

D - Incorrect: The switch expression selects a case; break exits the switch.

Primary note proof

Study-note section CP15

> break exits the nearest enclosing loop or switch. continue skips the rest of the current loop iteration: a for loop then runs its update; a while loop goes directly to its condition. return leaves the current function and may supply a result. In main it ends the program normally.

Study-note section CP13

> A traditional C++ switch selects using an integral or enumeration value. case labels are constant expressions. Execution starts at the matching label, then continues through subsequent statements until a break, return or the end. default handles an unmatched value but need not be the last physical label.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 13 | lines 30-35

Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Question sources

Computer Science MCQ practice collection · Question 272

Question 135 | AM0273

What is the output of the following code snippet?

```
int i = 0;
while (i < 5) {
cout << i << " ";
i++;
}
```

A. 0 1 2 3 4

B. 1 2 3 4 5

C. 5 4 3 2 1

D. 0 0 0 0 0

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer A: 0 1 2 3 4

The loop prints i before incrementing. Values 0,1,2,3,4 pass i<5; after the last increment i=5 fails the test (p.11). Module support: PDF p. 11, PDF p. 7, PDF p. 6.

Why each choice is right or wrong

A - Correct: Lists the five values printed before increments.

B - Incorrect: Starts after the increment and includes the excluded endpoint.

C - Incorrect: The counter increases from zero, not decreases.

D - Incorrect: i++ prevents i remaining zero.

Primary note proof

Study-note section CP14

while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Study-note section CP09

Prefix ++x changes x and yields the new value; postfix x++ yields the previous value and also increments x. Separate the value used in the expression from the stored value after it. Starting a=8,b=5, a++ + --b uses 8+4, gives 12, and leaves a=9,b=4. Starting x=1,y=10,z=-3, (x++*y--)+ ++z gives 8 and leaves x=2,y=9,z=-2.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 11 | lines 5-13

Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, | |) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 | | 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 6 | lines 4-15

6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Computer Science MCQ practice collection · Question 273

Question 136 | AM0274

What is the purpose of the cin object in C++?

A. To write output to the console.

B. To declare variables.

C. To perform mathematical calculations.

D. To read input from the user.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: To read input from the user.

Page 6 identifies cin as standard input and says it accepts information from the user. Module support: PDF p. 6.

Why each choice is right or wrong

A - Incorrect: This is cout.

B - Incorrect: A declaration requires a type and identifier.

C - Incorrect: Arithmetic operators calculate.

D - Correct: Matches the stated input role.

Primary note proof

Study-note section CP28

> Include <iostream> and qualify std::cin/std::cout or introduce the names explicitly. cout is the standard output stream; << inserts values. cin is standard input; >> extracts formatted values. std::endl inserts a newline and flushes. C++ names are case sensitive: Cout, Int and std: :cout are not equivalent valid spellings of cout, int and std::cout.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 6 | lines 4-15

> 6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Computer Science MCQ practice collection · Question 274

Computer Science MCQ practice collection · Question 275

Question 137 | AM0276

Under C++17, what does this fragment print? Assume cout is available.
int x=5;
cout<<x++<<" "<<++x;

A. 5 7

B. 5 5

C. 6 5

D. 6 6

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer A: A. 5 7

In C++17 the chained operator-form stream insertions are sequenced left to right. The postfix increment supplies the original value 5 to the first insertion and leaves x equal to 6. The later prefix increment changes x to 7 before supplying its value to the second insertion. The output is therefore 5 followed by a space and 7. Specifying the language version makes this a defined trace.

Why each choice is right or wrong

A - Correct: The postfix supplies 5 and the later prefix supplies 7.

B - Incorrect: The two increments prevent the second value from remaining 5.

C - Incorrect: The first postfix yields 5 rather than 6.

D - Incorrect: The first value is 5 and the second is 7, not two sixes.

Primary note proof

Study-note section CP09

> Precedence determines grouping, not a universal evaluation order. Avoid expressions that modify a variable multiple times without sequencing. In C++17 and later, chained operator-form stream insertions are sequenced left to right: with x=5, cout<<x++<<" "<<++x prints 5 7. Before C++17 this expression has undefined behavior. For the C++17 expression above, 5 6, 5 5, 6 5 and 6 6 are all incorrect outputs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, | |) Like the relational operators,

logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 8 | lines 12-23

{int a=10,b=3,c,d,e,f; c=a%b++; cout<<"c="<<c<<endl; b+=2; cout<<"b="<<b<<endl; d=++a+b; cout<<"d="<<d<<endl; e=a>b?a:b; e++; cout<<"e="<<e++<<endl; f=a+b+c+d+e; cout<<"f="<<f;

Question sources

Computer Science MCQ practice collection · Question 276

Question 138 | AM0277

What does this C++ fragment print? Assume cout is available and execution reaches the fragment normally.

```
int total=0;
for(int i=1;i<=4;++i){
if(i==3) break;
total+=i;
}
cout<<total<<" done";
```

A. No output; the program terminates when i reaches 3.

B. 3 done

C. 7 done

D. 10 done

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: B. 3 done

At i=1 and i=2, the condition is false and total becomes 1 and then 3. At i=3, break exits the nearest enclosing loop before the addition, so neither 3 nor 4 is added. Control continues after the loop and the output statement prints 3 done. A total of 7 would result from skipping only the i=3 iteration with continue; 10 would require completing every addition.

Why each choice is right or wrong

A - Incorrect: break exits the loop, so execution still reaches the following cout statement.

B - Correct: Only 1 and 2 are added before loop exit, and cout then appends the literal space and done.

C - Incorrect: 7 would include 4 after skipping 3, which describes continue rather than break.

D - Incorrect: 10 adds all four loop values and ignores the exit at i=3.

Primary note proof

Study-note section CP15

> break exits the nearest enclosing loop or switch. continue skips the rest of the current loop iteration: a for loop then runs its update; a while loop goes directly to its condition. return leaves the current function and may supply a result. In main it ends the program normally.

Study-note section CP14

> for(initialization; condition; update) executes initialization once, then condition → body → update → condition. A missing condition in for(;;) acts as true. A for loop is convenient for a known count but can also be condition controlled. To print indices 0 through n-1, use i=0; i<n; ++i. Starting i=0 with i<5 and i+=2 prints 0 2 4.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 13 | lines 30-35

> Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Question sources

Computer Science MCQ practice collection · Question 277

Question 139 | AM0279

What is the purpose of the return statement in a function?

A. To terminate the program.

B. To declare variables.

C. To pass values to another function.

D. To return a value from the function.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: To return a value from the function.

The sum function on p.17 returns x+y+z+w and its callers display the resulting values. This shows return delivering a value to the caller. Module support: PDF p. 17, PDF p. 14.

Why each choice is right or wrong

A - Incorrect: An ordinary function return need not terminate the program.

B - Incorrect: No variable is declared by return.

C - Incorrect: Arguments pass values into a function; the example return supplies its result back.

D - Correct: Matches the sum example and caller usage.

Primary note proof

Study-note section CP15

> break exits the nearest enclosing loop or switch. continue skips the rest of the current loop iteration: a for loop then runs its update; a while loop goes directly to its condition. return leaves the current function and may supply a result. In main it ends the program normally.

Study-note section CP17

> A function groups a named task. C++ definition: int add(int a,int b){return a+b;}. Call it with add(x,y);, without parameter type declarations at the call site. Parameters are names in the definition; arguments are supplied expressions or values. A void function produces no return value but can have side effects and return; to exit early.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 17 | lines 2-10

> int sum(int x, int y, int z = 0, int w = 0) { return (x + y + z + w); } int main() { cout << sum(10, 15) << endl; cout << sum(10, 15, 25) << endl; cout << sum(10, 15, 25, 30) << endl;

Computer Programming module | PDF page 14 | lines 21-33

> Function A function is a group of statements that structure our programs in a more modular way and executed when it is called from some point of the program. It is a block of code that performs a specific task. It is important to Dividing a complex problem into smaller chunks makes our program easy to understand and reusable. When we need function? - When you need to repeat the same process over and over in a program. - The function can be called many times but appears in the code once Function - Categories C++ functions can be divided into 2 categories: • pre-defined (standard function) / library function - Which is the definitions have been written and it is ready to be used. - User needs to include pre-defined header file (i.e., math.h, time.h,cmath) To use it: FunctionName (value) e.g., sqrt (4), floor (4.3)

Question sources

Computer Science MCQ practice collection · Question 279

Question 140 | AM0280

Which options are valid definitions of an ordinary initialized constant integer x in C++? Treat each option independently and assume no custom macros or aliases. Select all that apply.

A. const int x=5;

B. final int x;

C. int const x=5;

D. int x;

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer A, C: A. const int x=5;; C. int const x=5;

C++ permits const before or after the integer type specifier. Both const int x=5 and int const x=5 therefore define initialized constant integers. The initializer supplies the value required for the ordinary const object. final does not replace const in this declaration, and an unqualified int remains mutable. Treating the options independently avoids redeclaration conflicts while recognizing both equivalent const placements.

Why each choice is right or wrong

A - Correct: const int with an initializer defines the requested constant.

B - Incorrect: final is not the const qualifier in an ordinary integer declaration.

C - Correct: int const is equivalent to const int and has an initializer here.

D - Incorrect: An unqualified int variable remains mutable.

Primary note proof

Study-note section CP06

> Use int for whole numbers, float/double for fractional approximations, char for a character and bool for logical values in C++. Type sizes depend on the implementation; do not import Java widths into C++. short n, short int n and int short n denote the same signed short type. const int n=5 and int const n=5 are equivalent; an ordinary const int needs initialization. typedef creates an alias, not a distinct new type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 8 | lines 27-33

> Constant: A constant is any expression that has a fixed value like integer, float, character and string Eg. 120, 15.2, „A‟, “Hello There” Generally, we have two ways to define constant: 1. Using Const Key word Syntax: const type constant name=value; e.g. const float PI=3.14; 2. Using #define Keyword Syntax: #define constant name value e.g. #define PI 3.14

Question sources

Computer Science MCQ practice collection · Question 280

Question 141 | AM0281

Which option describes the defining execution and test behavior of a C++ do-while loop?

A. To execute a block exactly once regardless of its condition.

B. To perform mathematical calculations.

C. To iterate over a collection of elements.

D. To execute the body before its first test and repeat while the continuation condition is true.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: D. To execute the body before its first test and repeat while the continuation condition is true.

A do-while loop executes its body before the first condition test. It then repeats while its continuation condition is true and stops when that condition becomes false. Thus at least one body execution is guaranteed when the loop is reached, but exactly one is not guaranteed. Collection traversal and arithmetic can occur inside many constructs; neither describes this loop's distinctive control rule.

Why each choice is right or wrong

A - Incorrect: A do-while loop can repeat, so it is not limited to exactly one execution.

B - Incorrect: Arithmetic is possible in the body but is not the defining control behavior.

C - Incorrect: Collection traversal is an application rather than the defining test-and-repeat rule.

D - Correct: The body runs first and further iterations require a true continuation condition.

Primary note proof

Study-note section CP14

> while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 12 | lines 2-7

> The do-while loop The condition in the do-while is evaluated after the execution of statement instead of before, granting at least one execution of statement even if condition is never fulfilled. do {statement; }while (condition); First statement is executed and then condition is evaluated. If

the outcome of the latter is nonzero then the whole process is repeated. Otherwise, the loop is terminated.

Question sources

Computer Science MCQ practice collection · Question 281

Question 142 | AM0282

What is the output of the following code snippet?

```
int i = 0;
while (i < 5) {
cout << i << " ";
i += 2;
}
```

A. 0 1 2 3 4

B. 0 2 4

C. 0 1 2 3 4 5

D. 1 3 5

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 0 2 4

Starting at zero, printing precedes i+=2. The successive printed values are 0,2,4, followed by i=6 and a failed i<5 test. Page 7 explicitly illustrates compound addition assignment. Module support: PDF p. 11, PDF p. 7.

Why each choice is right or wrong

A - Incorrect: Would require unit increments.

B - Correct: Matches the traced states before each increment.

C - Incorrect: Includes intermediate values and 5 although the test excludes 5.

D - Incorrect: Starts at 1 instead of the specified 0.

Primary note proof

Study-note section CP14

while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Study-note section CP08

x+=y, x-=y, x*=y, x/=y and x%=y update x using the operation and conversion to its type. For simple variables, trace them as x=x op y. Starting p=q=n=8, p+=n gives 16 and p/=q gives 2. Starting x=10, x+=5 then x/=2 gives 7. x=x+x doubles; it does not generally add one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 11 | lines 5-13

Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science MCQ practice collection · Question 282

Question 143 | AM0283

What is the purpose of the continue statement in a loop?

A. To terminate the program.

B. To skip the remaining code within the loop and move to the next iteration.

C. To repeat the loop from the beginning.

D. To execute a specific case in a switch statement.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: To skip the remaining code within the loop and move to the next iteration.

Page 13 defines continue as skipping the remaining statements of the current iteration and transferring to the next iteration. Module support: PDF p. 13.

Why each choice is right or wrong

A - Incorrect: Does not terminate the program.

B - Correct: Exactly matches the cited definition.

C - Incorrect: Too broad: it proceeds through the loop continuation rules, rather than necessarily restarting all loop setup.

D - Incorrect: Does not select a switch case.

Primary note proof

Study-note section CP15

> break exits the nearest enclosing loop or switch. continue skips the rest of the current loop iteration: a for loop then runs its update; a while loop goes directly to its condition. return leaves the current function and may supply a result. In main it ends the program normally.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 13 | lines 30-35

> Using continue and break Statements - The break statement skips the rest of the statements in the loop‟s block, jumping immediately to the next statement outside of the loop. - The continue statement is similar. It also skips the rest of the statements in the loop‟s block, but instead of terminating the loop, it transfers execution to the next iteration of the loop. It continues the loop after skipping the remaining statements in its current iteration.

Question sources

Computer Science MCQ practice collection · Question 283

Question 144 | AM0284

What is the purpose of the modulo operator (%) in C++?

A. To perform addition.

B. To perform multiplication.

C. To perform division.

D. To calculate the remainder of integer division.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer D: To calculate the remainder of integer division.

The arithmetic list on p.7 includes %, and p.8 evaluates c=a%b++ for a=10,b=3; the previously inspected output c=1 demonstrates the remainder interpretation. Module support: PDF p. 7, PDF p. 8.

Why each choice is right or wrong

A - Incorrect: Addition is +.

B - Incorrect: Multiplication is *.

C - Incorrect: The quotient operation is /; the example produces a remainder.

D - Correct: 10 divided by 3 leaves the illustrated remainder 1.

Primary note proof

Study-note section CP08

> Multiplication, division and remainder have higher precedence than addition and subtraction; equal-precedence arithmetic groups left to right. Integer division truncates toward zero. For integers, a == (a/b)*b + a%b when the division is valid. % requires integral operands in C++; (8.0*3)%5 is invalid. Division by zero and signed integer overflow cannot be given ordinary numeric outputs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, | |) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 | | 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto

decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 8 | lines 12-23

{int a=10,b=3,c,d,e,f; c=a%b++; cout<<"c="<<c<<endl; b+=2; cout<<"b="<<b<<endl; d=++a+b; cout<<"d="<<d<<endl; e=a>b?a:b; e++; cout<<"e="<<e++<<endl; f=a+b+c+d+e; cout<<"f="<<f;

Question sources

Computer Science MCQ practice collection · Question 284

Question 145 | AM0285

What is the output of the following code snippet?

```
int x = 10;
int y = 3;
cout << x / y;
```

A. 3.33333

B. 3

C. 3.0

D. 3.33333...

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 3.

Both x and y have type int, so CP08's integer-division rule applies: 10/3 truncates toward zero to 3. The stream prints that integer value. No floating conversion occurs before division, and no decimal formatting is requested, so neither a fractional quotient nor the literal output 3.0 is correct.

Why each choice is right or wrong

A - Incorrect: 3.33333 is a fractional approximation, but division of these int operands discards the fractional part.

B - Correct: Integer division gives 3, and cout prints the resulting integer without a decimal suffix.

C - Incorrect: 3.0 is not the printed representation of the int quotient in this code.

D - Incorrect: An indefinitely repeating decimal is not produced by integer division or this stream insertion.

Primary note proof

Study-note section CP08

Multiplication, division and remainder have higher precedence than addition and subtraction; equal-precedence arithmetic groups left to right. Integer division truncates toward zero. For integers, a == (a/b)*b + a%b when the division is valid. % requires integral operands in C++; (8.0*3)%5 is invalid. Division by zero and signed integer overflow cannot be given ordinary numeric outputs.

Study-note section CP28

Read a full line with getline(cin,text). After formatted numeric extraction, consume the pending newline before a line read when required. Input failure must be tested before trusting a value. Console output formatting matters: by default a double storing an exact 4 may print 4 rather than 4.0. Text, spaces and newlines are part of an output trace.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, | |) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 | | 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science MCQ practice collection · Question 285

Question 146 | AM0286

Which listed C++ construct computes a number raised to a power, using the standard mathematical library?

A. ^

B. **

C. pow()

D. ^=

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: C. pow()

C++ supplies pow as a mathematical library function for exponentiation. It is a function call rather than a built-in exponentiation operator. The caret denotes bitwise XOR, and ** is not a C++ power operator; ^= consequently does not provide exponentiation assignment. With the required mathematical declarations available, pow is the construct matching the requested operation, so the intended C key remains valid.

Why each choice is right or wrong

A - Incorrect: ^ is bitwise XOR rather than exponentiation.

B - Incorrect: ** is not a C++ exponentiation operator.

C - Correct: The standard-library pow function computes powers.

D - Incorrect: ^= uses the bitwise operation, not exponentiation.

Primary note proof

Study-note section CP30

> C++ has no built-in exponentiation operator: ^ is bitwise XOR and ** is not a power operator. To compute b^e for a nonnegative integer e, start r=1 and repeat r*=b exactly e times; b=4,e=3 gives 64. Large powers and factorials can overflow fixed-width types; specify a valid input range.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 2-7

> - ceil : smallest integral value not less than x - fmod : floating-point remainder function - log : logarithm - pow : Raise a number by a power. - sin : The sine of an integer. - sqrt : Square root of a number.

Question sources

Computer Science MCQ practice collection · Question 286

Question 147 | AM0287

What is the output of the following code snippet?

```
double result = sqrt(16);
cout << result;
```

A. 2

B. 4

C. 8

D. 16

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 4

Page 15 defines sqrt as square root. Since 4 multiplied by 4 is 16, sqrt(16) is 4. Its worked example also displays sqrt(25) as 5, supporting the plain integral-looking output format. Module support: PDF p. 15, PDF p. 15.

Why each choice is right or wrong

A - Incorrect: 2 squared is 4, not 16.

B - Correct: 4 squared equals 16.

C - Incorrect: 8 squared is 64.

D - Incorrect: 16 is the argument, not its square root.

Primary note proof

Study-note section CP30

> Use <cmath> for sqrt, pow, floor and ceil; integer abs is also provided by <cstdlib>. sqrt(16)=4; pow(3,4)=81; abs(-7)=7. ceil(x) is the least integer-valued result >=x; floor(x) the greatest <=x. Thus ceil(-2.7)=-2 and floor(-2.7)=-3; truncation toward zero differs from floor for negatives.

Study-note section CP28

> Read a full line with getline(cin,text). After formatted numeric extraction, consume the pending newline before a line read when required. Input failure must be tested before trusting a value. Console output formatting matters: by default a double storing an exact 4 may print 4 rather than 4.0. Text, spaces and newlines are part of an output trace.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 2-7

- ceil : smallest integral value not less than x - fmod : floating-point remainder function - log : logarithm - pow : Raise a number by a power. - sin : The sine of an integer. - sqrt : Square root of a number.

Computer Programming module | PDF page 15 | lines 10-21

Example #include<iostream> #include <cmath> using namespace std; int main() {double x=-25; float y=36.5; int z=25; cout<<"absolute value:"<<fabs(x)<<endl; cout<<floor(y)<<endl; cout<<ceil(y)<<endl; cout<<sqrt(z);

Question sources

Computer Science MCQ practice collection · Question 287

Question 148 | AM0288

Which header file should be included to use mathematical functions in C++?

A. <iostream>

B. <cmath>

C. <string>

D. <iomanip>

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: <cmath>

The module math-function example on p.15 includes <cmath> before using fabs, floor, ceil and sqrt. This directly supports B among these headers. Module support: PDF p. 15.

Why each choice is right or wrong

A - Incorrect: The example also includes iostream for streams; the math header is cmath.

B - Correct: This is the demonstrated math header.

C - Incorrect: string is not the math header in the cited example.

D - Incorrect: iomanip is not the math header demonstrated here.

Primary note proof

Study-note section CP30

> Use <cmath> for sqrt, pow, floor and ceil; integer abs is also provided by <cstdlib>. sqrt(16)=4; pow(3,4)=81; abs(-7)=7. ceil(x) is the least integer-valued result >=x; floor(x) the greatest <=x. Thus ceil(-2.7)=-2 and floor(-2.7)=-3; truncation toward zero differs from floor for negatives.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 10-21

> Example #include<iostream> #include <cmath> using namespace std; int main() {double x=-25; float y=36.5; int z=25; cout<<"absolute value:"<<fabs(x)<<endl; cout<<floor(y)<<endl; cout<<ceil(y)<<endl; cout<<sqrt(z);

Question sources

Computer Science MCQ practice collection · Question 288

Question 149 | AM0289

What is the output of the following code snippet?

```
int x = 5;
int y = 2;
cout << x % y;
```

A. 0

B. 1

C. 2

D. 3

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 1

Using the remainder operation illustrated by c=10%3 on p.8, 5=2*2+1 leaves remainder 1. Module support: PDF p. 8.

Why each choice is right or wrong

A - Incorrect: Zero would require exact divisibility.

B - Correct: The remainder after two whole groups of 2 is 1.

C - Incorrect: 2 is the whole-number quotient, not the remainder.

D - Incorrect: A remainder of 3 would exceed the positive divisor 2.

Primary note proof

Study-note section CP08

> Multiplication, division and remainder have higher precedence than addition and subtraction; equal-precedence arithmetic groups left to right. Integer division truncates toward zero. For integers, a == (a/b)*b + a%b when the division is valid. % requires integral operands in C++; (8.0*3)%5 is invalid. Division by zero and signed integer overflow cannot be given ordinary numeric outputs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 8 | lines 12-23

> {int a=10,b=3,c,d,e,f; c=a%b++; cout<<"c="<<c<<endl; b+=2; cout<<"b="<<b<<endl; d=++a+b; cout<<"d="<<d<<endl; e=a>b?a:b; e++; cout<<"e="<<e++<<endl; f=a+b+c+d+e; cout<<"f="<<f;

Question sources

Computer Science MCQ practice collection · Question 289

Question 150 | AM0290

What is the output of the following code snippet?

```
double result = pow(3, 4);
cout << result;
```

A. 7

B. 12

C. 27

D. 81

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: 81

The p.15 definition of pow gives 3 raised to 4: 3*3*3*3=81. Module support: PDF p. 15.

Why each choice is right or wrong

A - Incorrect: 7 is addition of the arguments.

B - Incorrect: 12 is multiplication of the arguments.

C - Incorrect: 27 uses exponent 3 instead of 4.

D - Correct: 81 is the fourth power.

Primary note proof

Study-note section CP30

> Use <cmath> for sqrt, pow, floor and ceil; integer abs is also provided by <cstdlib>. sqrt(16)=4; pow(3,4)=81; abs(-7)=7. ceil(x) is the least integer-valued result >=x; floor(x) the greatest <=x. Thus ceil(-2.7)=-2 and floor(-2.7)=-3; truncation toward zero differs from floor for negatives.

Study-note section CP28

> Read a full line with getline(cin,text). After formatted numeric extraction, consume the pending newline before a line read when required. Input failure must be tested before trusting a value. Console output formatting matters: by default a double storing an exact 4 may print 4 rather than 4.0. Text, spaces and newlines are part of an output trace.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 2-7

- ceil : smallest integral value not less than x - fmod : floating-point remainder function - log : logarithm - pow : Raise a number by a power. - sin : The sine of an integer. - sqrt : Square root of a number.

Question sources

Computer Science MCQ practice collection · Question 290

Question 151 | AM0291

What is the purpose of the abs() function in C++?

A. To calculate the absolute value of a number.

B. To calculate the average of two numbers.

C. To round a floating-point number to the nearest integer.

D. To compare two numbers for equality.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: To calculate the absolute value of a number.

Page 14 distinguishes fabs for a floating-point absolute value and abs for an integer. Thus abs supplies the integer absolute value. Module support: PDF p. 14.

Why each choice is right or wrong

A - Correct: Matches the stated integer absolute-value role.

B - Incorrect: No averaging role is given.

C - Incorrect: Rounding is not the abs role; floor and ceil are separate.

D - Incorrect: Equality is a relational operation, not abs.

Primary note proof

Study-note section CP30

> Use <cmath> for sqrt, pow, floor and ceil; integer abs is also provided by <cstdlib>. sqrt(16)=4; pow(3,4)=81; abs(-7)=7. ceil(x) is the least integer-valued result >=x; floor(x) the greatest <=x. Thus ceil(-2.7)=-2 and floor(-2.7)=-3; truncation toward zero differs from floor for negatives.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 14 | lines 34-36

> The Standard Math Library - fabs : absolute value of floating point number, #include<cstdlib> to call abs() for integer number -floor: largest integral value not greater than x

Question sources

Computer Science MCQ practice collection · Question 291

Question 152 | AM0292

What is the output of the following code snippet?

```
int x = -7;
cout << abs(x);
```

A. -7

B. 7

C. 0

D. Undefined

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 7

Page 14 identifies abs for integers. The absolute value is distance from zero, so -7 gives 7. Module support: PDF p. 14.

Why each choice is right or wrong

A - Incorrect: Keeps the negative sign instead of taking magnitude.

B - Correct: Magnitude of -7 is 7.

C - Incorrect: Only zero has zero magnitude.

D - Incorrect: No undefined input is present for this small integer under the stated function assumption.

Primary note proof

Study-note section CP30

> Use <cmath> for sqrt, pow, floor and ceil; integer abs is also provided by <cstdlib>. sqrt(16)=4; pow(3,4)=81; abs(-7)=7. ceil(x) is the least integer-valued result >=x; floor(x) the greatest <=x. Thus ceil(-2.7)=-2 and floor(-2.7)=-3; truncation toward zero differs from floor for negatives.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 14 | lines 34-36

> The Standard Math Library - fabs : absolute value of floating point number, #include<cstdlib> to call abs() for integer number -floor: largest integral value not greater than x

Question sources

Computer Science MCQ practice collection · Question 292

Question 153 | AM0293

What is the purpose of the ceil() function in C++?

A. To calculate the ceiling value of a floating-point number.

B. To calculate the factorial of an integer.

C. To round a floating-point number down to the nearest integer.

D. To perform bitwise complement on an integer.

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: To calculate the ceiling value of a floating-point number.

Page 15 defines ceil as the smallest integral value not less than x. This is the ceiling operation. Module support: PDF p. 15.

Why each choice is right or wrong

A - Correct: Matches the exact ceiling definition.

B - Incorrect: Factorial is unrelated to the cited definition.

C - Incorrect: Rounding downward corresponds to floor, not ceil.

D - Incorrect: No bitwise operation appears in the definition.

Primary note proof

Study-note section CP30

> Use <cmath> for sqrt, pow, floor and ceil; integer abs is also provided by <cstdlib>. sqrt(16)=4; pow(3,4)=81; abs(-7)=7. ceil(x) is the least integer-valued result >=x; floor(x) the greatest <=x. Thus ceil(-2.7)=-2 and floor(-2.7)=-3; truncation toward zero differs from floor for negatives.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 2-7

> - ceil : smallest integral value not less than x - fmod : floating-point remainder function - log : logarithm - pow : Raise a number by a power. - sin : The sine of an integer. - sqrt : Square root of a number.

Question sources

Computer Science MCQ practice collection · Question 293

Question 154 | AM0294

Which of the following data types in C++ can store fractional values?

A. int

B. char

C. float

D. bool

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: C. float

The float type represents fractional numeric approximations, so it can retain a fractional component. int represents whole numbers; converting a fractional value to it does not preserve that component. char represents a character, and bool represents the logical values true and false. These different value domains make float the unique listed type intended for fractional quantities, without implying that every real number is represented exactly.

Why each choice is right or wrong

A - Incorrect: An int stores a whole-number result rather than a retained fractional component.

B - Incorrect: A char represents a character, not the listed fractional numeric type.

C - Correct: A float represents fractional numeric approximations.

D - Incorrect: A bool represents logical true/false values rather than fractional magnitudes.

Primary note proof

Study-note section CP06

> Use int for whole numbers, float/double for fractional approximations, char for a character and bool for logical values in C++. Type sizes depend on the implementation; do not import Java widths into C++. short n, short int n and int short n denote the same signed short type. const int n=5 and int const n=5 are equivalent; an ordinary const int needs initialization. typedef creates an alias, not a distinct new type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 24-33

> 5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Computer Programming module | PDF page 15 | lines 15-21

```
{double x=-25; float y=36.5; int z=25; cout<<"absolute value:"<<fabs(x)<<endl;
cout<<floor(y)<<endl; cout<<ceil(y)<<endl; cout<<sqrt(z);
```

Question sources

Computer Science MCQ practice collection · Question 294

Question 155 | AM0295

What is the result of the expression 5 / 2 in C++?

A. 2

B. 2.5

C. 2.0

D. 2.25

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: The integer value 2.

Both literals in 5/2 are integers. CP07 states that the integer quotient is 2, and even assigning that already-computed quotient to double would not restore the lost fraction. The question asks for the expression's result, so the integer 2 is the precise choice rather than a floating representation or fractional quotient.

Why each choice is right or wrong

A - Correct: Two integer operands produce the integer quotient 2 after truncating the fractional part.

B - Incorrect: 2.5 requires floating division, which these uncast integer literals do not perform.

C - Incorrect: 2.0 is a floating representation, while this expression itself has an integer result.

D - Incorrect: 2.25 follows neither integer truncation nor the ordinary mathematical quotient of five divided by two.

Primary note proof

Study-note section CP07

With int x=5,y=2, x/y is 2. Assigning that quotient to double gives 2.0, not 2.5. static_cast<double>(x)/y performs floating division and yields 2.5. Converting an in-range fractional floating value to int truncates toward zero. Cast before the operation if a fractional result is needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science MCQ practice collection · Question 295

Question 156 | AM0296

What is the output of the following code snippet?

```
int x = 5;
int y = 2;
double result = static_cast<double>(x) / y;
cout << result;
```

A. 2

B. 2.5

C. 2.0

D. 2.25

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 2.5.

static_cast<double>(x) converts 5 to a floating value before the division takes place. CP07 explicitly gives this form as yielding 2.5 when x=5 and y=2. The result therefore retains the fractional part; storing it in double does not first perform an integer quotient.

Why each choice is right or wrong

A - Incorrect: Casting x before division prevents the integer truncation that would have produced 2.

B - Correct: The converted floating operand makes the quotient 2.5, which is stored and printed.

C - Incorrect: 2.0 would result from first computing the integer quotient and then converting it, unlike this code.

D - Incorrect: The floating quotient of 5 by 2 is 2.5, not 2.25.

Primary note proof

Study-note section CP07

With int x=5,y=2, x/y is 2. Assigning that quotient to double gives 2.0, not 2.5. static_cast<double>(x)/y performs floating division and yields 2.5. Converting an in-range fractional floating value to int truncates toward zero. Cast before the operation if a fractional result is needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 28 | lines 25-41

void pointers - Void pointer point to any data type Example: void *ptr; int x=5; Ptr=&x; #include <iostream> using namespace std; int main() {int a=10; float b=20; void*ptr; ptr=&a; cout<<*(static_cast<int*>(ptr))<<endl; //cout<<*((int*)ptr) ptr=&b; cout<<*((float*)ptr)<<endl;

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science MCQ practice collection · Question 296

Question 157 | AM0297

What is the value of x after executing the following code snippet?

```
int x = 10;
x += 5;
x /= 2;
```

A. 5

B. 7

C. 8

D. 10

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: x becomes 7.

The initial x is 10. x+=5 updates it to 15; x/=2 then performs integer division because x is int, truncating 15/2 toward zero to 7. CP08 provides this exact sequence and result. The updates happen in order, so the second operation acts on 15 rather than the original 10.

Why each choice is right or wrong

A - Incorrect: Five divides the original value by two and ignores the preceding addition of five.

B - Correct: The sequence gives 10+5=15, then integer 15/2=7.

C - Incorrect: Integer division truncates 7.5 to 7 rather than rounding it to 8.

D - Incorrect: Both compound assignments update x; it does not retain its initial value of 10.

Primary note proof

Study-note section CP08

> x+=y, x-=y, x*=y, x/=y and x%=y update x using the operation and conversion to its type. For simple variables, trace them as x=x op y. Starting p=q=n=8, p+=n gives 16 and p/=q gives 2. Starting x=10, x+=5 then x/=2 gives 7. x=x+x doubles; it does not generally add one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science MCQ practice collection · Question 297

Question 158 | AM0298

What is the result of the following expression?

bool result = (5 > 3) && (10 < 8);

A. true

B. false

C. compilation error

D. runtime error

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: false

Page 7 lists relational and logical operators and represents true and false as 1 and 0. Here 5>3 is true and 10<8 is false; logical AND requires both, so result is false. Module support: PDF p. 7.

Why each choice is right or wrong

A - Incorrect: Would require both comparisons to be true.

B - Correct: The second comparison is false, making the conjunction false.

C - Incorrect: The operators are listed module constructs, and the declaration follows the taught type-name form.

D - Incorrect: No runtime failure is implied by either simple comparison.

Primary note proof

Study-note section CP10

> Relational and equality operators test values; && and || combine truth values; ! negates. In C++, zero converts to false and a nonzero number to true. Streams display bool as 0 or 1 unless boolalpha is set. With a=3,b=5, cout<<(a==b) prints 0.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

> Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto

decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Question sources

Computer Science MCQ practice collection · Question 298

Question 159 | AM0299

What is the output of the following code snippet?

```
int x = 3;
int y = 5;
cout << (x == y);
```

A. 1

B. 0

C. true

D. false

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 0

Page 7 explicitly says relational operators evaluate to 1 for true or 0 for false. Since 3 and 5 differ, x==y evaluates to 0; cout displays that result (p.6). Module support: PDF p. 7, PDF p. 6.

Why each choice is right or wrong

A - Incorrect: 1 would mean equality is true.

B - Correct: 0 represents the false equality.

C - Incorrect: The equality is not true.

D - Incorrect: This names the logical result but differs from the numeric representation taught by the module.

Primary note proof

Study-note section CP10

Relational and equality operators test values; && and || combine truth values; ! negates. In C++, zero converts to false and a nonzero number to true. Streams display bool as 0 or 1 unless boolalpha is set. With a=3,b=5, cout<<(a==b) prints 0.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 6 | lines 4-15

6. Communication through console cout (the standard output stream) o used to display information for the user Syntax: cout<<object; • Object can be variable or literals Example: int x=5; cout<<x cout<< “Hello there”; cin(the standard input stream): o used to accept information from the user. Syntax: cin>>variable name; Eg. cin>>x;

Question sources

Computer Science MCQ practice collection · Question 299

Question 160 | AM0300

What is the value of result after executing the following code snippet?

```
int a = 3;
int b = 2;
int result = pow(a, b);
```

A. 6

B. 9

C. 8

D. 5

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 9

The module defines pow as raising a number to a power (p.15). With a=3 and b=2, it computes 3*3=9; this whole-number value is the intended integer result. Module support: PDF p. 15.

Why each choice is right or wrong

A - Incorrect: 6 multiplies base and exponent.

B - Correct: 9 is 3 squared.

C - Incorrect: 8 is 2 cubed, reversing arguments.

D - Incorrect: 5 adds the arguments.

Primary note proof

Study-note section CP30

> Use <cmath> for sqrt, pow, floor and ceil; integer abs is also provided by <cstdlib>. sqrt(16)=4; pow(3,4)=81; abs(-7)=7. ceil(x) is the least integer-valued result >=x; floor(x) the greatest <=x. Thus ceil(-2.7)=-2 and floor(-2.7)=-3; truncation toward zero differs from floor for negatives.

Study-note section CP07

> With int x=5,y=2, x/y is 2. Assigning that quotient to double gives 2.0, not 2.5. static_cast<double>(x)/y performs floating division and yields 2.5. Converting an in-range fractional floating value to int truncates toward zero. Cast before the operation if a fractional result is needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 15 | lines 2-7

> - ceil : smallest integral value not less than x - fmod : floating-point remainder function - log : logarithm - pow : Raise a number by a power. - sin : The sine of an integer. - sqrt : Square root of a number.

Question sources

Computer Science MCQ practice collection · Question 300

Question 161 | AM0301

What is the output of the following code snippet?

```
int x = -5;
cout << abs(x);
```

A. -5

B. 5

C. 0

D. compilation error

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 5

Page 14 identifies integer abs. The magnitude of -5 is 5, so cout displays 5. Module support: PDF p. 14.

Why each choice is right or wrong

A - Incorrect: Retains the negative sign.

B - Correct: Gives the magnitude of -5.

C - Incorrect: -5 is not zero.

D - Incorrect: With the required declarations, this is the integer abs usage described in the module.

Primary note proof

Study-note section CP30

> Use <cmath> for sqrt, pow, floor and ceil; integer abs is also provided by <cstdlib>. sqrt(16)=4; pow(3,4)=81; abs(-7)=7. ceil(x) is the least integer-valued result >=x; floor(x) the greatest <=x. Thus ceil(-2.7)=-2 and floor(-2.7)=-3; truncation toward zero differs from floor for negatives.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 14 | lines 34-36

> The Standard Math Library - fabs : absolute value of floating point number, #include<cstdlib> to call abs() for integer number -floor: largest integral value not greater than x

Question sources

Computer Science MCQ practice collection · Question 301

Question 162 | AM0302

What is the value of result after executing the following code snippet?

```
double pi = 3.14159;
int result = floor(pi);
```

A. 3

B. 4

C. 3.14

D. 3.14159

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer A: 3

Page 14 defines floor as the largest integral value not greater than x. For 3.14159 that value is 3, because 3<=3.14159<4. Module support: PDF p. 14.

Why each choice is right or wrong

A - Correct: 3 is the greatest integer not exceeding pi.

B - Incorrect: 4 is greater than pi and would be a ceiling.

C - Incorrect: 3.14 remains fractional and is not floor.

D - Incorrect: 3.14159 is the original argument, not its floor.

Primary note proof

Study-note section CP30

> Use <cmath> for sqrt, pow, floor and ceil; integer abs is also provided by <cstdlib>. sqrt(16)=4; pow(3,4)=81; abs(-7)=7. ceil(x) is the least integer-valued result >=x; floor(x) the greatest <=x. Thus ceil(-2.7)=-2 and floor(-2.7)=-3; truncation toward zero differs from floor for negatives.

Study-note section CP07

> With int x=5,y=2, x/y is 2. Assigning that quotient to double gives 2.0, not 2.5. static_cast<double>(x)/y performs floating division and yields 2.5. Converting an in-range fractional floating value to int truncates toward zero. Cast before the operation if a fractional result is needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 14 | lines 34-36

> The Standard Math Library - fabs : absolute value of floating point number, #include<cstdlib> to call abs() for integer number -floor: largest integral value not greater than x

Question sources

Computer Science MCQ practice collection · Question 302

Question 163 | AM0303

What is the output of the following code snippet?

```
int x = 4;
int y = 2;
cout << (x % y);
```

A. 0

B. 1

C. 2

D. 4

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer A: 0

The remainder operation illustrated on p.8 gives 4%2=0 because 4=2*2+0. Module support: PDF p. 8.

Why each choice is right or wrong

A - Correct: Exact divisibility leaves zero remainder.

B - Incorrect: A remainder of 1 would represent an odd dividend.

C - Incorrect: 2 is the quotient and cannot be the remainder for positive divisor 2.

D - Incorrect: 4 is the original dividend, not its remainder.

Primary note proof

Study-note section CP08

> Multiplication, division and remainder have higher precedence than addition and subtraction; equal-precedence arithmetic groups left to right. Integer division truncates toward zero. For integers, a == (a/b)*b + a%b when the division is valid. % requires integral operands in C++; (8.0*3)%5 is invalid. Division by zero and signed integer overflow cannot be given ordinary numeric outputs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 8 | lines 12-23

```
{int a=10,b=3,c,d,e,f; c=a%b++; cout<<"c="<<c<<endl; b+=2; cout<<"b="<<b<<endl; d=++a+b;
cout<<"d="<<d<<endl; e=a>b?a:b; e++; cout<<"e="<<e+++<<endl; f=a+b+c+d+e; cout<<"f="<<f;
```

Question sources

Computer Science MCQ practice collection · Question 303

Question 164 | CS0086

What is the output of the following C++ fragment code? int x = 15, y = 2; int z = x / y; cout << z;

A. 7.5

B. 7

C. 3

D. 12.5

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: 7.

x and y are int variables with values 15 and 2. Their division therefore produces integer 7 before assignment to z, as CP08 specifies truncation toward zero. z also has type int and cout prints 7. None of the fractional or unrelated numerical options follows from this expression.

Why each choice is right or wrong

A - Incorrect: 7.5 is the mathematical fractional quotient, but these int operands produce an integer quotient.

B - Correct: The integer quotient 15/2 truncates toward zero to 7, which z stores and cout prints.

C - Incorrect: Three is not the quotient obtained by dividing the stated values 15 and 2.

D - Incorrect: 12.5 is neither the integer quotient nor the fractional quotient of 15 divided by 2.

Primary note proof

Study-note section CP08

> Multiplication, division and remainder have higher precedence than addition and subtraction; equal-precedence arithmetic groups left to right. Integer division truncates toward zero. For integers, a == (a/b)*b + a%b when the division is valid. % requires integral operands in C++; (8.0*3)%5 is invalid. Division by zero and signed integer overflow cannot be given ordinary numeric outputs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 7 | lines 5-23

Assignment Operators (=) Example: int x=5; or used with a combination of other operators‟ example, x+=2;//x=x+2, x=y=10; Arithmetic Operators (+, -, *, /, %) E.g. Assume int X=9, Y=2 X+Y, Y-X, X%Y Relational operators (==,!=, <, <=,>,>=) Relational operators evaluate to 1 (representing the true outcome) or 0 (representing the false outcome). Assume int X=9, Y=2 X==Y, X!=Y, X>Y 'A' < 'F' // gives 1 (is like 65 < 70) Logical operators (! Negation, &&, ||) Like the relational operators, logical operators evaluate to 1 or 0. Example: 5 < 6 || 6 < 5 // gives 1 In general, any nonzero value can be used to represent the logical true, whereas only zero represents the logical false !12 // gives 0 10 && 5 // gives 1 Increment/decrement Operators The auto increment (++) and auto decrement (--) operators provide a convenient way of, respectively, adding and subtracting 1 from a numeric variable. Both operators can be used in prefix and postfix form.

Computer Programming module | PDF page 5 | lines 24-33

5. Data Types, Variables, and Constants Variable: - Variable is a symbolic name for a memory location in which data can be stored. - Attribute of Variables: o A type, which is, established when the variable is defined (e.g., integer, float, character) o A value, which can be changed by assigning a new value to the variable Variable Declaration - Declaring a variable means defining (creating) a variable. - Syntax: data type variable name; e.g. int num; - We can create More Than One Variable at a Time e.g., int num1, num2, num3;

Question sources

Computer Science Exit Exam practice collection · Question 86

Question 165 | CS0087

Writing comments in between our C++ program makes our program _____.

A. Portable

B. Maintainable

C. Reliable

D. Efficient

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer B: Maintainable

Page 5 says comments explain aspects of a program for human readers and are ignored by the compiler. Explanations help a maintainer understand and change code, so maintainable is the best fit. Module support: PDF p. 5.

Why each choice is right or wrong

A - Incorrect: Human explanatory text does not establish cross-platform compatibility.

B - Correct: Explanatory text helps people understand code during maintenance.

C - Incorrect: Comments alone do not establish correctness or reliable execution.

D - Incorrect: Ignored comments do not establish runtime efficiency.

Primary note proof

Study-note section CP05

> Comments use // to the line end or /* ... */ for a block. They explain intent and assumptions for maintenance; ordinary comments do not execute or automatically improve speed, reliability or portability. main is the conventional entry function name, not a C++ keyword.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 5 | lines 18-20

> 4. Comments A comment is a piece of descriptive text which explains some aspect of a program. Program comments are totally ignored by the compiler and are only intended for human readers.

Question sources

Computer Science Exit Exam practice collection · Question 87

Question 166 | CS0088

What is the output of the following C++ fragment code? int data[10] = {5,10,15,20,25,30,35,40,45,50}; int i = 0; while (i <= 10) { if(data[i] % 10 == 0) { cout << data[i] << " "; i = i + 2; } }

A. 5 15 25 35 45 55

B. 10 20 30 40 50

C. 5 10 15 20 25 30 35 40 45 50

D. No output

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: No output

Arrays are zero indexed (p.19), so initially data[0]=5. Its remainder on division by 10 is 5, not zero, so the if body is skipped. The only increment of i lies inside that skipped body. Therefore i stays zero, i<=10 remains true, and no cout is ever reached. The loop runs indefinitely with no output; it never reaches the potentially invalid index 10. Module support: PDF p. 19, PDF p. 11, PDF p. 8, PDF p. 9.

Why each choice is right or wrong

A - Incorrect: No print occurs at the first iteration or later, so this odd-value sequence cannot appear.

B - Incorrect: The loop never advances to data[1]=10.

C - Incorrect: No array traversal occurs because i never changes.

D - Correct: Correct: nothing is printed, and the loop does not terminate.

Primary note proof

Study-note section CP16

> For n=1; n<=18; n+=2, print n unless n%7==0 then break: output is 1 3 5. A while(a>b) with fixed a=6,b=5 never changes its true condition. In data[10]={5,10,...,50}, i=0, advancing i only if data[i]%10==0 causes no progress because data[0]=5. It prints nothing and loops indefinitely; changing <=10 to <10 alone does not repair progress.

Study-note section CP22

> A built-in C++ array has a fixed number of same-type elements in contiguous storage: int a[4]={10,20,30,40};. Valid indices are 0 through 3. a[4] is outside the array; accessing it has undefined behavior. Java checks bounds and throws ArrayIndexOutOfBoundsException. Do not predict a C++ out-of-range value from nearby memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 19 | lines 4-29

Arrays - an array is a collection of fixed-size number of elements of the same type stored sequentially in memory. - An individual element of an array is identified by its own unique index (or subscript). - The number of elements must be an integer. - The size of the array is referred to as its dimension - Array in C++ is Zero indexed. Declaring of one-dimensional Array Syntax: elementstype arrayname [size]; → You can use constants such as macro definitions or const ints as the sizes of arrays. Example: int num[5]; string fruit[3]; #define numstudents 100 or const int numstudents=100; double Score[numstudents] ; double Score[numstudents] ; Initialization of an Array - Arrays can be initialized at declaration time in two different ways: Syntax: elementstype arrayname[size] = {list each value separated by comma}; E.g int num[5]={1,2,3,4,5}; elementstype arrayname[ ] = { list with any number of values }; E.g int num[]={1,2,3,4,5}; string fruit[]={“Apple”,“orange”, “Mango”}; Accessing Array Elements - Each element could be accessed using an index into the array - When accessing an array, the index given must be a positive integer from 0 to n-1, where n is the dimension of the array.

Computer Programming module | PDF page 11 | lines 5-13

Iteration/Looping Statement: Iteration is the repetition of a statement or block of statements in a program. - Loops have as objective to repeat a statement a certain number of times or while a condition is fulfilled. - C++ has three iteration statements: the while statement, the do. While statement, and the for statement. The while loop: Its format is: while (expression) {statement;} First expression (called the loop condition) is evaluated. If the outcome is nonzero then statement (called the loop body) is executed and the whole process is repeated. Otherwise, the loop is terminated.

Computer Programming module | PDF page 8 | lines 12-23

{int a=10,b=3,c,d,e,f; c=a%b++; cout<<"c="<<c<<endl; b+=2; cout<<"b="<<b<<endl; d=++a+b; cout<<"d="<<d<<endl; e=a>b?a:b; e++; cout<<"e="<<e++<<endl; f=a+b+c+d+e; cout<<"f="<<f;

Computer Programming module | PDF page 9 | lines 9-21

Conditional Statements A. The if Statement Syntax: if (expression) statement; First expression is evaluated. If the outcome is nonzero (true) then statement is executed. Otherwise, nothing happens. if-else statement Syntax: if (expression) statement1; else statement2; First expression is evaluated. If the outcome is nonzero (true) then statement1 is executed. Otherwise, statement2 is executed.

Question sources

Computer Science Exit Exam practice collection · Question 88

Question 167 | CS0092

After a compiler reports a syntax error and before rebuilding, which action directly helps locate and repair its cause?

A. Rewrite the entire program

B. Ignore the error and continue execution

C. Use a debugger tool

D. Check the code for typos or incorrect syntax

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer D: D. Check the code for typos or incorrect syntax

A syntax error violates the language grammar, so inspecting the relevant tokens and nearby declarations is the direct diagnostic step. The note recommends starting with the first useful compiler diagnostic, repairing the code and rebuilding afterward. Rewriting everything is unnecessary, ignoring the diagnostic leaves the defect in place, and a debugger is useful after a runnable program exists. The revised timing asks for inspection rather than the final verification step.

Why each choice is right or wrong

A - Incorrect: A complete rewrite is unnecessary for locating a local grammar mistake.

B - Incorrect: Ignoring a compiler error leaves the cause unresolved.

C - Incorrect: A debugger is useful after a runnable program exists.

D - Correct: Inspecting nearby tokens and declarations helps locate and correct grammar mistakes.

Primary note proof

Study-note section CP32

> A syntax error violates grammar; a type error can be syntactically well formed but rejected during semantic checking. A runtime error occurs during execution; a logic error gives the wrong behavior. A bug is a defect; debugging locates and corrects it. Read the first useful compiler diagnostic, inspect nearby tokens and declarations, repair and rebuild; a debugger is useful after a runnable program exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Programming module | PDF page 4 | lines 21-30

> The compilation process of C++ program consists of the following steps. • edit source code → compile → link → execute (Editor) (Compiler) (Linker) (Loader) I. Program source code is entered into a file using a text editor. II. After the code has been entered, a compiler program is started that translates the source into an object code file. III. The object code file is linked with other

object code files that come with the compiler and an executable file (or program) is created. IV. In order to execute the program, a program called the loader copies the executable file into the memory of the computer and sends an execute command to the CPU.

Computer Programming module | PDF page 2 | lines 2-18

1. Introduction to programming Computer programming is the process of writing, testing, debugging/troubleshooting, and maintaining the source code of computer programs. Programs are the instructions that tells the computer what to do .so, writing computer programs means writing instructions, that will make the computer follow and run a program based on those instructions. We are going to write computer programs to solve problems. Problem Solving Problem: gap between the existing and the desired or proposed situation where problem solving will try to fill this gap. Problem solving: The process of transforming the description of a problem into the solution by using our knowledge of the problem-solving strategies, techniques and tools. Problem solving techniques • Analyze the problem: Carefully understand the problem • Outline the problem requirements • Does program require user interaction? • Does program manipulate data? • What is the output? • Design steps (algorithm) to solve the problem

Question sources

Computer Science Exit Exam practice collection · Question 92

Question 168 | NS0278

What is the difference between a compiler and an interpreter? Which is Java?

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer: Compilers translate representations; interpreters execute them. Java uses bytecode compilation and JVM interpretation/JIT.

CP04 says javac produces bytecode, then a JVM interprets or compiles it at runtime. A compiler translates rather than necessarily producing a complete native executable, while an interpreter executes a program representation. Java therefore cannot be reduced to exclusively compiled or interpreted.

Primary note proof

Study-note section CP04

> In the conventional C++ toolchain: source → preprocessing → compilation proper → assembly → object files → linking → executable → loading → execution. Preprocessing handles directives such as #include; compilation may emit assembly; the assembler emits machine instructions in an object file. The linker resolves external symbols and combines object files and libraries. The loader places an executable and required components in memory and starts it.

Study-note section CP04

> GCC documents the first four stages explicitly. A compiler need not emit a complete executable or even native machine code. Java javac produces bytecode, which a JVM interprets or compiles at runtime. Interpreters execute a program representation; “one source line at a time” is a beginner approximation. Compilation uses multiple passes, so “whole program in one step” is not literally universal.

Question sources

Computer Programming review questions · Question 1

Question 169 | NS0279

What is the difference between = (assignment) and == (comparison)? Why is this a common mistake?

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer: = assigns a value; == compares values for equality.

CP06 describes assignment as replacing a variable's value; CP10 describes equality as a value test. Confusing them can change state instead of testing it. In C++ a numeric assigned result can also convert to a condition, allowing an unintended assignment to alter control flow.

Primary note proof

Study-note section CP-S05

> In C++, simple assignment changes its left-hand object, and the assignment expression refers to that updated object. When an ordinary numeric assignment is used as a condition, the newly assigned value is converted to truth: zero is false and nonzero is true. Equality testing with == instead produces a Boolean result without assigning either operand. Confusing the two can both change program state and choose an unintended branch.

Study-note section CP10

> Relational and equality operators test values; && and || combine truth values; ! negates. In C++, zero converts to false and a nonzero number to true. Streams display bool as 0 or 1 unless boolalpha is set. With a=3,b=5, cout<<(a==b) prints 0.

Question sources

Computer Programming review questions · Question 2

Question 170 | NS0280

What is the difference between while and do-while loops? When would you use do-while?

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer: while tests first; do-while runs once before testing. Use do-while for a mandatory first menu/input attempt.

CP14 allows a while body to execute zero times because its condition is checked first. A do-while body executes before its first check, making it suitable for a menu or input attempt that must happen at least once.

Primary note proof

Study-note section CP14

> while(condition) checks before each body execution and may run zero times. do {body;} while(condition); runs the body once before checking; it repeats while the condition remains true. Use it when a menu or input attempt must occur at least once.

Question sources

Computer Programming review questions · Question 3

Question 171 | NS0281

What happens if you try to access array index array[5] when the array has 5 elements (indices 0-4)?

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer: Index 5 is out of bounds: Java throws; C++ access has undefined behavior.

Five elements occupy indices 0 through 4, so array[5] is outside the array. CP22 distinguishes Java's bounds exception from C++ undefined behavior; an adjacent-memory value must not be guessed. The question does not specify a language, so both consequences are stated.

Primary note proof

Study-note section CP22

> A built-in C++ array has a fixed number of same-type elements in contiguous storage: int a[4]={10,20,30,40};. Valid indices are 0 through 3. a[4] is outside the array; accessing it has undefined behavior. Java checks bounds and throws ArrayIndexOutOfBoundsException. Do not predict a C++ out-of-range value from nearby memory.

Question sources

Computer Programming review questions · Question 4

Question 172 | NS0282

Why do we use functions? What are the benefits?

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer: Functions decompose named tasks, enable reuse, reduce duplication and simplify understanding/testing.

CP17 defines a function as grouping a named task and explains that reuse reduces duplication while independent functions are easier to understand and test. Decomposition localizes responsibilities and supports maintenance through those clearer units.

Primary note proof

Study-note section CP17

> A function groups a named task. C++ definition: int add(int a,int b){return a+b;}. Call it with add(x,y);, without parameter type declarations at the call site. Parameters are names in the definition; arguments are supplied expressions or values. A void function produces no return value but can have side effects and return; to exit early.

Question sources

Computer Programming review questions · Question 5

Question 173 | NS0283

What is the difference between pass-by-value and pass-by-reference?

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer: C++ value parameters copy; reference parameters alias caller variables. Java passes copied values, including references.

CP18 makes a value parameter a separate initialized object, while int& aliases the caller variable and can change it directly. Passing a pointer copies an address and can change its target without rebinding the caller's pointer. Java copies object references by value.

Primary note proof

Study-note section CP18

> A C++ value parameter is a separate object initialized from the argument. In func(int a,int b){a*=5;b+=5;return a+b;}, calling func(9,3) returns 53 and leaves the caller values 9 and 3 unchanged.

Study-note section CP18

> An int& parameter aliases the caller variable, so swap(int& a,int& b) changes both originals. An int* parameter copies an address; *p=value changes the pointed-to object, but p=anotherAddress only changes the local pointer. An array parameter int a[] is adjusted to int*: elements can change, and the length must be passed separately. Java always passes by value, including a copy of an object reference; it does not pass caller variables by reference.

Question sources

Computer Programming review questions · Question 6

Question 174 | NS0301

What does this C++ fragment print? Assume cout is available.

```
int a=6,b=8;
int x=2,y=4;
int c(x>y ? (a--,x) : (b--,y));
cout<<"a="<<a;
cout<<" b="<<b;
cout<<" c="<<c;
```

A. a=6 b=5 c=4

B. a=6 b=7 c=5

C. a=8 b=6 c=5

D. a=6 b=7 c=4

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: D. a=6 b=7 c=4

The condition x>y compares 2 with 4 and is false. Only the false branch is evaluated. Its comma expression first decrements b from 8 to 7 and then yields y, which is 4, for initializing c. The true branch does not execute, so a remains 6. With valid stream spelling, comma operators and ordinary string delimiters, the output is a=6 b=7 c=4.

Why each choice is right or wrong

A - Incorrect: b decreases once from 8 to 7, not to 5.

B - Incorrect: The comma expression yields y=4, not 5.

C - Incorrect: The false branch leaves a=6 and produces b=7,c=4.

D - Correct: Only b decrements; the branch yields y=4 and a remains 6.

Primary note proof

Study-note section CP11

> condition ? yes : no evaluates the condition and exactly one alternative. The comma operator evaluates its left expression, then its right, and yields the right value. Do not confuse it with commas separating function arguments.

Study-note section CP09

> Prefix ++x changes x and yields the new value; postfix x++ yields the previous value and also increments x. Separate the value used in the expression from the stored value after it. Starting a=8,b=5, a++ + --b uses 8+4, gives 12, and leaves a=9,b=4. Starting x=1,y=10,z=-3, (x++*y--)+ ++z gives 8 and leaves x=2,y=9,z=-2.

Question sources

Computer Science Exit Exam · 2018 · Question 19

Question 175 | NS0325

Assume that you are computer science expert and you need to develop large programs within short period of time. Finally, you have planned to provide portable program to the concerned body. Which type of programming language is preferable for the above scenario?

A. Low level

B. Machine level

C. High level

D. Assembly

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer C: A high-level language, with portability constraints checked.

CP03 says high-level source is easier to read and often portable after recompilation if platform assumptions are avoided. That fits rapid development of a large program and a portable result better than instruction-set-dependent machine or assembly code. Portability remains conditional on the target environment; the answer is a preference, not an unconditional guarantee.

Why each choice is right or wrong

A - Incorrect: Low-level code retains instruction-set details and does not best match the stated readability/development and portability goals.

B - Incorrect: Machine-level code depends directly on the instruction set, opposing the requested portable-source preference.

C - Correct: High-level source offers the note's readability and conditional portability advantages for the stated scenario.

D - Incorrect: Assembly depends on the instruction set and is not the preferred abstraction for these development goals.

Primary note proof

Study-note section CP03

> Machine and assembly code depend on an instruction set. High-level source is easier to read and often portable after recompilation, provided it avoids platform assumptions. People can write both assembly and high-level programs. Python, Perl and PHP are commonly called scripting languages; C++ is conventionally compiled. These labels describe common use, not an absolute technical prohibition.

Question sources

Computer Science Exit Exam · 2018 · Question 51

Question 176 | NS0346

Which one of the following identifier name is INVALID in C++ programming language?

A. delete

B. for_cpp

C. jan2025

D. Hello

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer A: delete is a reserved keyword.

delete is a C++ keyword and cannot be used as an ordinary identifier. for_cpp is not the keyword for, and jan2025 and Hello use letters followed only by permitted letters or digits.

Why each choice is right or wrong

A - Correct: delete is a reserved C++ keyword and therefore invalid as an ordinary identifier.

B - Incorrect: for_cpp is a distinct permitted name containing letters and underscore, rather than the keyword for.

C - Incorrect: jan2025 begins with a letter and contains only permitted letters and trailing digits.

D - Incorrect: Hello contains permitted letters and is not a reserved keyword identified by the question.

Primary note proof

Study-note section CP05

> In the basic ASCII identifier rules, start with a letter or underscore; continue with letters, digits or underscores. Do not use a keyword. C++ is case sensitive: exitExam and ExitExam differ; delete and float are keywords, but FLOAT is a legal identifier. variable_1234 is valid; 4thInput, T-3 and names containing spaces are not single identifiers. Avoid reserved underscore patterns in application code.

Question sources

Computer Science Exit Exam · 2018 · Question 79

Question 177 | NS0358

Which of the following statement about function overloading is TRUE?

A. Overload function have same name but different parameters/type

B. Overload function is not supported

C. Overload function must have different return type

D. Overload function must have the same number of parameters

Hint

Use the cited language rules to separate the purpose of each construct from its implementation details.

Answer A: Overloads share a name and differ in their parameter lists.

CP19 requires a shared name with differences in parameter count or types/order. A states that relationship. Return type alone does not distinguish overloads, and neither different return types nor equal parameter counts are mandatory. Thus the other statements either deny supported overloading or impose restrictions contradicted here.

Why each choice is right or wrong

A - Correct: This states the shared-name and differing-parameter-list relationship required by CP19.

B - Incorrect: CP19 explicitly describes supported C++ function overloading, contradicting this blanket denial.

C - Incorrect: Different return types are not required and return type alone cannot distinguish overloads.

D - Incorrect: Parameter count may differ between overloads, so equal counts are not mandatory.

Primary note proof

Study-note section CP19

> Overloads share a name but differ in parameter count or types/order. Return type alone cannot distinguish them. Choose a candidate using argument types, then required conversions; report ambiguity if there is no unique best match. int sum(int x,int y,int z=0) allows two or three arguments; defaults occupy trailing positions in ordinary declarations.

Question sources

Computer Science Exit Exam · 2018 · Question 96

Fundamental of Database Systems

Questions & Worked Answers

226 questions + 2 source variants

Question 1 | Q00021

Which one of the following is not correct when mapping ER diagrams to relational tables?

A. Multi-valued attributes are mapped to new tables

B. Foreign key are used to map relationships to create relation between tables

C. Table names are directly mapped from entity names

D. Composite attribute are mapped as it is

Hint

Apply the ER mapping rule for the given attribute or entity type.

Answer D: Composite attribute are mapped as it is

Atomic relational values exclude storing an undecomposed composite attribute. Relation names need not differ from entity names.

Why each choice is right or wrong

A - Incorrect: Multivalues are separated to remove repeating groups (PDF 12, 17.)

B - Incorrect: Foreign-key mapping is explicitly described (PDF 12, 17.)

C - Incorrect: Naming a relation after its entity is compatible with the mapping (PDF 12, 17.)

D - Correct: Keeping a composite value unchanged violates atomicity (PDF 12, 17.)

Primary note proof

Study-note section FDB10

> Create one relation per strong entity type, preserve its simple attributes and select a candidate key. Expand composite attributes into their useful simple components; do not copy a composite ER structure unchanged into a flat atomic design. Derived values can usually be calculated, avoiding stale stored copies.

Study-note section FDB10

> Table names may follow entity names; naming does not invalidate a mapping.

Study-note section FDB10

> Map each multivalued attribute to a separate relation containing the owner key and one value.

Related reading (context, not primary note proof)

Study-note section FDB11

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 12 | lines 9-24

> 1. For each strong entity type E, create a relation R that includes all the attributes of E. 2. Choose one of the key attributes of E as the primary key for R. For each weak entity type W with owner entity type E (which is a strong entity), create a relation R & include all attributes of _W‘ as attributes of R. Also, include the primary key attribute of the relation that correspond to the owner entity type as foreign key attributes of R . 3. For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. Foreign Key approach: Choose one of the relations-say S-and include a foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S. 4. For each binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type. Include the primary key of the relation T as foreign key in S that represents the other entity type participating in R. Include any attributes of the 1:N relation type as attributes of S. 5. For each n-ary relationship type R, where n>2, create a new relationship S to represent R. Include the primary keys of the relations that represent the participating entity types as foreign key attributes in S. Also include any attributes of the n-ary relationship type as attributes of S.

Fundamental of Database Systems module | PDF page 17 | lines 2-9

> First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF).

Question sources

MOE Computer Science Exit Exam · 2016 · Question 21

Question 2 | Q00038

In DBMS-independent logical relational design, before vendor-specific refinement, which statement is NOT correct?

A. It consider a particular DBMS

B. Normalization is undertaken in logical phase

C. It is based on specific data model

D. It is independent from physical considerations

Hint

Identify the design phase from its inputs and intended deliverable.

Answer A: A. It consider a particular DBMS

Logical design maps the conceptual model into relations, keys and constraints and checks normalization. It therefore uses a particular data model while separating storage decisions from logical structure. Under the stated DBMS-independent methodology, considering a particular vendor's DBMS is the false claim. Normalization, a chosen model and independence from physical storage are all consistent with this phase.

Why each choice is right or wrong

A - Correct: False in the stated phase: particular-DBMS implementation choices are deferred.

B - Incorrect: True: normalization checks the mapped relations.

C - Incorrect: True: relational logical design uses a specific data model.

D - Incorrect: True: file layouts and indexes belong to physical design.

Primary note proof

Study-note section FDB05

> First collect business facts, required transactions, reports, volumes, privacy rules and quality targets. Conceptual design turns them into an ER/EER model. Logical design maps the conceptual model to relations, keys and constraints and checks normalization. Physical design chooses file organization, indexes and storage. Implementation creates and loads the actual objects; testing checks requirements.

Study-note section FDB05

> Whether logical design is DBMS-dependent depends on the stated design methodology.

Study-note section FDB05

> Logical relational design is tied to a data model but can remain independent of a particular DBMS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 21-22

> Logical database design (Data model mapping) The conceptual design is translated into internal model which includes mapping of all objects i.e

Fundamental of Database Systems module | PDF page 19 | lines 6-7

> data, a physical database design is needed. The goal is to create a design for storing and retrieving data that will provide adequate performance and insure database integrity, security, and

Question sources

MOE Computer Science Exit Exam · 2016 · Question 38

Question 3 | Q00040

Which SQL command would you use if you want to specify constraints to be applied on tuples?

A. GRANT

B. SELECT

C. REVOKE

D. CHECK

Hint

Read each SQL constraint's scope and enforcement rule.

Answer D: CHECK

CHECK verifies criteria on attribute values within rows; GRANT and REVOKE control permission, and SELECT retrieves data.

Why each choice is right or wrong

A - Incorrect: Permission grant is not a tuple predicate (PDF 49, 54.)

B - Incorrect: Retrieval is not constraint specification (PDF 49, 54.)

C - Incorrect: Permission removal is not a tuple predicate (PDF 49, 54.)

D - Correct: CHECK is explicitly shown verifying criteria (PDF 49, 54.)

Primary note proof

Study-note section FDB16

> Example: CREATE TABLE Enrollment (StudentID INTEGER NOT NULL REFERENCES Student(StudentID), CourseID INTEGER NOT NULL REFERENCES Course(CourseID), Grade INTEGER CHECK (Grade BETWEEN 0 AND 100), PRIMARY KEY (StudentID,CourseID)); Each column declaration gives its name then type, with commas between declarations. CHECK is a constraint clause, not a standalone GRANT-like statement. POSITIVE is not a general SQL constraint keyword; CHECK (Amount > 0) expresses positivity.

Study-note section FDB16

> A CHECK can pass when its result is unknown, so add NOT NULL if missing values must be forbidden.

Study-note section FDB26

> DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 49 | lines 19-22

● CREATE TABLE table-name (attr2 attr-type NOT NULL, forbids NULL values attrN attr-type CHECK (attrN = UPPER(attrN) verifies that the data meets certain criteria attrO attr-type DEFAULT default_value); Supplies default values

Fundamental of Database Systems module | PDF page 54 | lines 6-18

2.2.3 Authorization Rules ● Most current DBMS permit the DBA to define —access permissions‖ on a table by table basis (at least) using the GRANT and REVOKE SQL commands ● Some systems permit finer grained authorization (most use GRANT and REVOKE on variant views 2.2.4 Privileges in SQL GRANT <privileges> ON <object> TO <users> [WITH GRANT OPTION] • <privileges> is a list of SELECT <columns>, INSERT <columns>, DELETE, and UPDATE <columns>, or simply ALL

Fundamental of Database Systems module | PDF page 26 | lines 10-11

7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

MOE Computer Science Exit Exam · 2016 · Question 40

Question 4 | Q00041

Which data management approach allows application programs to define and manage their own data?

A. Manual approach

B. Database approach

C. Hierarchical approach

D. Traditional file based approach

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer D: Traditional file based approach

The file approach lacks the database approach separation between programs and data; application-owned data corresponds to traditional file processing. B reverses that distinction.

Why each choice is right or wrong

A - Incorrect: Manual records do not describe application-owned files (PDF 5, 6.)

B - Incorrect: Program/data insulation contradicts the stem (PDF 5, 6.)

C - Incorrect: Hierarchical is a model rather than this processing distinction (PDF 5, 6.)

D - Correct: Traditional file processing matches the contrast (PDF 5, 6.)

Primary note proof

Study-note section FDB01

A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB01

Traditional file applications define their own formats and access procedures.

Study-note section FDB04

Data abstraction hides irrelevant details; program-data insulation separates data definitions from application code.

Related reading (context, not primary note proof)

Study-note section FDB02

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 17-25

1.2 Database System versus File System The traditional file processing system is file-directory structure supported by a conventional operating system. A file system organization of data lucks a number of major features of a database system, such as Data redundancy and inconsistency, Difficulty in accessing data, Data isolation, Integrity problems, Atomicity problems, Concurrent access anomalies. 1.3 Characteristics of the Database Approach The main characteristics of the database approach are Self-describing nature, Data abstraction and Insulation between programs and data, Support of multiple views of the data, sharing of data and multiuser transaction processing.

Fundamental of Database Systems module | PDF page 6 | lines 2-4

Data abstraction and Insulation between programs and data- Data Abstraction is a process of hiding unwanted or irrelevant details from the end user. It provides a different view and helps in achieving data independence which is used to enhance the security of data.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 41

Question 5 | Q00043

Which database constraints are specified in the data definition language?

A. Implicit or inherent model based constraints

B. Semantic constraints

C. Application based constraints

D. Schema based constraints

Hint

Match the validity requirement to the appropriate constraint definition.

Answer D: Schema-based constraints.

The question asks how constraints are specified, rather than which business meaning they express. FDB15 distinguishes constraints inherent in a data model from constraints explicitly declared in schema DDL. That second category is schema-based. Semantic or application rules may require triggers or application code when ordinary DDL cannot express them.

Why each choice is right or wrong

A - Incorrect: Inherent model constraints follow from the data model rather than explicit schema declarations.

B - Incorrect: Semantic rules express application meaning and may exceed what ordinary schema DDL can enforce.

C - Incorrect: Application-based constraints may require program logic rather than declarations in the database definition.

D - Correct: Schema-based constraints are explicitly specified in data definition language declarations.

Primary note proof

Study-note section FDB15

> Inherent constraints come from the data model; schema-based constraints are declared through the DDL; semantic/application rules may need triggers or application logic when DDL cannot express them. Integrity improves data validity; authorization controls who may access it. Uniqueness and domain validation are both integrity mechanisms. If a Department primary key is NULL and Manager=107 lacks a referenced employee, entity integrity certainly fails and referential integrity also fails if that foreign key is declared. One INSERT can violate multiple constraints.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 5-6

> has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language).

Fundamental of Database Systems module | PDF page 49 | lines 17-18

> • This is now increasing handled by the database. In Oracle, for example, when defining a table you can specify:

Question sources

MOE Computer Science Exit Exam · 2016 · Question 43

Question 6 | Q00080

Which SQL command removes an existing relation's table object, including its definition?

A. Delete

B. Drop

C. Remove

D. UPDATE

Hint

Classify each SQL command by the kind of change it performs.

Answer B: B. Drop

Removing a relation object means removing its definition as well as its contents. DROP TABLE performs that operation. DELETE affects rows while retaining the table, and UPDATE changes values within rows. REMOVE is not the SQL table-removal command. The distinction is between schema removal and operations on data in an existing schema.

Why each choice is right or wrong

A - Incorrect: DELETE removes rows and retains the table definition.

B - Correct: DROP TABLE removes the table object and its definition.

C - Incorrect: REMOVE is not the SQL command for dropping a table.

D - Incorrect: UPDATE changes existing row values.

Primary note proof

Study-note section FDB26

> DELETE removes rows and keeps the table; DROP TABLE removes the table object; ALTER changes its definition. TRUNCATE empties a table using implementation-specific mechanisms. PostgreSQL allows transactional rollback of TRUNCATE, so no rollback information is retained is not a universal SQL property. SQL Server permits DROP DATABASE Sales, suppliers; that comma-list syntax is vendor-specific. SQLite's .databases lists attached names/files in its command-line shell; .database is not the documented command.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 8-9

> Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME

Fundamental of Database Systems module | PDF page 26 | lines 11-11

> Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

MOE Computer Science Exit Exam · 2016 · Question 80

Question 7 | Q00106

Which of the following statement best describe relational data model?

A. It is a database model that allows multiple records to be linked to the same owner file.

B. It structures data in a tree like structure using parent to child relationship.

C. It is a database model to manage data as tuples grouped into relations

D. It is a model that use concepts such as entities, attributes, and relationships

E. Describe how data is stored in the computer, representing information such as record structures, record ordering, and access paths.

Hint

Compare the data models by how they organize records and relationships.

Answer C: It is a database model to manage data as tuples grouped into relations

A relational database consists of relations with tuple rows and attribute columns.

Why each choice is right or wrong

A - Incorrect: Owner-file links do not define tuple relations (PDF 13, 7.)

B - Incorrect: A tree is not the stated relational representation (PDF 13, 7.)

C - Correct: Tuple rows in relations match the definition (PDF 13, 7.)

D - Incorrect: Entities and relationships describe conceptual ER modeling (PDF 13, 7.)

E - Incorrect: Storage details define the physical model (PDF 13, 7.)

Primary note proof

Study-note section FDB02

> A data model supplies concepts for structure, relationships, constraints and operations. Conceptual ER models describe entities and relationships. Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents. Object models represent objects with identity, state and behavior. A physical model specifies record layouts and access paths. These are different representation choices, not stages of one query.

Study-note section FDB03

> Degree counts columns; cardinality counts rows.

Study-note section FDB03

> A tuple/row/record contains related attribute values; an attribute/column/field describes a property.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 13 | lines 19-20

> The relational model represents the database as a collection of relations. A table is represented with columns and rows. Each table is an entity in the ER diagram. Each row is known as a tuple.

Fundamental of Database Systems module | PDF page 7 | lines 12-13

> 2. Low-level or physical data models provide concepts that describe the details of how data is stored on the computer storage media. Concepts provided by physical data models are generally meant

Question sources

Addis Ababa University mock exam · Question 8

Question 8 | Q00107

Department(Dept_Name, Manager) has primary key Dept_Name and foreign key Manager referencing Employee(Emp_ID). Employee contains IDs 101-105 only. Consider INSERT INTO Department (Dept_Name, Manager) VALUES (NULL, 107); Which named integrity rules are violated? Select all that apply. Treat key integrity as uniqueness and domain integrity as type/allowed-value checks.

A. Domain constraint

B. Key constraint

C. Entity Integrity constraint

D. Referential Integrity constraint

Hint

Match the validity requirement to the appropriate constraint definition.

Answer C, D: C. Entity Integrity constraint; D. Referential Integrity constraint

The inserted row has two independent defects. Dept_Name is the primary key, so NULL violates entity integrity. Manager is a non-NULL foreign key, yet employee 107 is absent, violating referential integrity. These are separate requirements and the same insertion fails both. Under the question's taxonomy, neither a duplicate-key violation nor an incompatible-domain value is being asserted.

Why each choice is right or wrong

A - Incorrect: The stated domain checks are satisfied; primary-key NULL is classified separately.

B - Incorrect: No duplicate key value is inserted; this category tests uniqueness.

C - Correct: NULL in the primary key violates entity integrity.

D - Correct: Manager 107 has no referenced employee and violates referential integrity.

Primary note proof

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 48 | lines 12-13

> ◦ Cannot be NULL iv. Referential Integrity

Fundamental of Database Systems module | PDF page 48 | lines 14-15

> • A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key

Question sources

Addis Ababa University mock exam · Question 9

Question 9 | Q00108

Assume atomic attribute values. Consider R(A,B,C,D) with functional dependencies F = {C→D, C→B, B→C}. Which statement about R is true?

A. The relation R is in second normal form

B. The relation R is in third normal form

C. The candidate key for R is BC

D. The relation R is in Boyce-Codd normal form

E. None of the other statements is true

Hint

Derive every candidate key before testing dependency-based normal forms.

Answer E: None of the other statements is true

Every candidate key must contain A because no dependency derives it. Both AB and AC determine all attributes and are minimal; BC determines only B, C and D. The nonprime attribute D depends on B or C alone, a proper subset of either composite key, so 2NF fails. The dependency C→D also violates 3NF and BCNF because C is not a superkey and D is nonprime. None of the other statements is true.

Why each choice is right or wrong

A - Incorrect: False: D depends on a proper subset of a candidate key, violating 2NF.

B - Incorrect: False: C→D has a nonsuperkey determinant and a nonprime dependent attribute, violating 3NF.

C - Incorrect: False: $(BC)^+ = \{B,C,D\}$, so BC cannot determine A and is not a key.

D - Incorrect: False: C determines D but is not a superkey, violating BCNF.

E - Correct: Correct: each of the other four claims is false.

Primary note proof

Study-note section FDB18

> For R(A,B,C,D), let C→D, C→B and B→C. A never appears on a right-hand side, so every key needs A. $(AB)^+$ gives ABCD through B→C→D; $(AC)^+$ also gives ABCD. AB and AC are minimal keys. BC cannot derive A and is not a key. A, B and C are prime; D is nonprime.

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Study-note section FDB21

> For every nontrivial FD X→A, 3NF permits it when X is a superkey or A is prime. BCNF requires X to be a superkey.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 15 | lines 5-17

> A functional dependency is a constraint between two attributes or sets of attributes of entities in a database. A functional dependency is denoted by X → Y, between two sets of attributes X and Y that are subsets of relation R. The constraint is that, the values of Y component of a tuple in r depend on, or are determined by the values of X component. Alternatively, the values of X component of a tuple uniquely (or functionally) determine the values of Y component. 4.2 Types of Functional Dependency Partial Dependency - If an attribute which is not a member of the key is dependent on some part of the key (if we have composite/candidate key) then that attribute is partially functionally dependent on the key. Let {A, B} is the key set and C is no key attribute, then if {A, B} → C and [B → C or A → C] holds. We say C is partially functionally dependent on {A, B}. Partial dependency occurs when a non-prime attribute is functionally dependent on part of a candidate/composite key.

Fundamental of Database Systems module | PDF page 17 | lines 10-15

> Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

> Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Addis Ababa University mock exam · Question 10

Question 10 | Q00109

Which schema of the three-schema Architecture describe the structure of the whole database?

A. Internal schema

B. Conceptual schema

C. External schema

D. schema

Hint

Trace which schema levels a proposed change affects.

Answer B: Conceptual schema

The conceptual schema describes the whole database while internal describes storage and external describes each user view.

Why each choice is right or wrong

A - Incorrect: Internal covers physical storage (PDF 8.)

B - Correct: Conceptual explicitly covers the whole database (PDF 8.)

C - Incorrect: External covers a user subset (PDF 8.)

D - Incorrect: Generic schema does not identify the required level (PDF 8.)

Primary note proof

Study-note section FDB04

> The external level gives each user group a selected view. The conceptual level describes the whole logical database: entity types, relationships, data types and constraints. The

internal/physical level describes files, record placement and access paths. These are abstraction levels, distinct from client/server deployment tiers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 25-30

2.3 Three-schema Architecture and Data Independence The goal of the three-schema architecture is to separate the user applications from the physical database. In this architecture, schemas can be defined at the following three levels: 1. The internal level has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.

Fundamental of Database Systems module | PDF page 8 | lines 2-7

2. The conceptual level has a conceptual schema, which describes the structure of the whole database for users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. 3. The external or view level includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

Question sources

Addis Ababa University mock exam · Question 11

Question 11 | Q00110

Which of the following is the disadvantage of a database approach?

A. Data consistency

B. Sharing of data

C. Improved data integrity

D. Increased concurrency

E. Enforcement of standards

F. Complexity

G. None

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer F: Complexity is a database-approach disadvantage.

FDB01 identifies complexity and administration costs as possible disadvantages of a DBMS. Consistency, sharing, integrity and centrally supported standards are benefits. Increased concurrency means useful concurrent access here; only uncontrolled concurrent access causes anomalies. Thus complexity is the sole listed disadvantage, even though database management can improve the other listed properties.

Why each choice is right or wrong

A - Incorrect: Data consistency is improved by shared definitions and controlled updates, making it a benefit.

B - Incorrect: Sharing data is a purpose and benefit of the database approach.

C - Incorrect: Improved integrity means better enforcement of valid data and is a benefit.

D - Incorrect: Concurrent access is useful; uncontrolled interference, rather than increased concurrency itself, causes problems.

E - Incorrect: Central management supports consistent definitions and standards, which is a benefit.

F - Correct: A DBMS introduces administrative and technical complexity that can outweigh benefits for small applications.

G - Incorrect: A listed disadvantage exists: the complexity of database management.

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB01

> Concurrent access itself is useful; uncontrolled concurrent access causes anomalies.

Study-note section FDB01

> Separate files can repeat facts, disagree after updates, isolate useful data and make shared integrity or concurrent updates difficult.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 23-24

> The main characteristics of the database approach are Self-describing nature, Data abstraction and Insulation between programs and data, Support of multiple views of the data, sharing of data and

Fundamental of Database Systems module | PDF page 117 | lines 6-7

i. Complexity- The data replication , failure recovery , network management ...make the system more complex than the central DBMSs

Question sources

Addis Ababa University mock exam · Question 12

Question 12 | Q00111

Which of the following is a correct statement about relation? (Choose all that applies)

A. Values of each columns is a repeating group/array

B. Values in a column can be from different domain

C. A relation is in the INF

D. Two rows in a relational table can be identical

E. The sequence of columns and rows is insignificant

F. None

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer C, E: A relation is in the INF; The sequence of columns and rows is insignificant

Read INF as 1NF. Relations have atomic cells rather than repeating groups; the relational representation is independent of tuple sequence. The column sequence likewise does not alter named attributes. Distinct tuples and typed attribute domains rule out A,B,D.

Why each choice is right or wrong

A - Incorrect: Repeating arrays violate atomicity (PDF 14, 17.)

B - Incorrect: An attribute has a constrained domain (PDF 14, 17.)

C - Correct: Atomic relation values satisfy 1NF (PDF 14, 17.)

D - Incorrect: Duplicate tuples conflict with relation uniqueness (PDF 14, 17.)

E - Correct: Presentation order does not change named tuple values (PDF 14, 17.)

F - Incorrect: C and E already supply valid statements (PDF 14, 17.)

Primary note proof

Study-note section FDB03

A relation is a set of tuples, represented as a table. A tuple/row/record contains related attribute values; an attribute/column/field describes a property. Degree counts columns; cardinality counts rows. Classical relations have distinct tuples, atomic cells and domains for attributes; row order has no meaning. Column order has no logical meaning when attributes are identified by name. SQL tables can contain duplicate rows unless constrained.

Study-note section FDB20

1NF requires atomic values in the chosen relational domains and no repeating groups in cells.

Study-note section FDB20

A mapped ER design that correctly splits composite and multivalued structures gives ordinary 1NF tables; ER drawing alone guarantees no higher form.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 14 | lines 4-20

Properties of Relations • Name of the relation is distinct from all other relations. • Each relation cell contains exactly one atomic (single) value. • Each attribute contains a distinct name. • Each tuple has no duplicate value. • Order of tuple can have a different sequence. 3.8 Relational Constraints Relational constraints in DBMS are referred to conditions which must be present for a valid relation. These Relational constraints in DBMS are derived from the rules in the system that the database represents. There are many types of Constraints in DBMS. Domain Constraints - Every domain (cell in the table or attribute value) must contain atomic values (smallest indivisible units). We perform data type check here, which means we need to assign a data type to a column and limit the values that it can contain. Key Constraints- These are called uniqueness constraints since it ensures that every tuple in the relation should be unique. An attribute that can uniquely identify a tuple in a relation is called the key or primary key of the table. The value of the attribute for different tuples in the relation has to be unique. Primary keys must be unique and cannot be null.

Fundamental of Database Systems module | PDF page 17 | lines 2-9

First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF).

Question sources

Addis Ababa University mock exam · Question 13

Question 13 | Q00112

Which of the following statement is correct about executing the following SQL query on the employee database given in question 2? (Choose all that applies)
SELECT Emp_ID, First_Name, Last_Name
FROM Employee E, Department D
WHERE Department <> Sales' and AVG (salary) > 35000 and
E.Department=D.Dept_Name
GROUP BY Department;

A. the query will execute successfully and will obtain all departments with an average salary higher than 35,000

B. the query will not execute because Department attribute in Group by statement was not included in the select statement list

C. the query will not execute because we can't include aggregate function in the where statement

D. the query will execute successfully and will obtain employees of all departments with an average salary higher than 35,000

Hint

Apply grouping before filtering groups and computing aggregate output.

Answer C: AVG(salary) cannot be used directly in WHERE.

The query places AVG(salary) > 35000 in WHERE. FDB32 requires ordinary aggregate predicates in HAVING. Omitting Department from SELECT is not itself an error. Selecting ungrouped employee details is another defect unless functional dependence is established. The query therefore does not successfully produce either claimed result.

Why each choice is right or wrong

A - Incorrect: The aggregate predicate in WHERE prevents this query from successfully producing department averages.

B - Incorrect: A grouping column need not appear in SELECT; that omission is not the stated error.

C - Correct: WHERE filters individual rows before grouping, so the ordinary AVG predicate belongs in HAVING.

D - Incorrect: The invalid aggregate predicate prevents execution; employee columns also require valid grouping or functional dependence.

Primary note proof

Study-note section FDB32

> COUNT(*) counts rows, COUNT(x) counts non-NULL x values; SUM and AVG summarize non-NULL numeric values. SQL's usual average function is AVG, not AVERAGE. GROUP BY creates one group per distinct grouping value; HAVING tests aggregate groups after WHERE has filtered individual rows.

Study-note section FDB32

> The grouping column need not itself appear in SELECT; check the functional dependence of other selected columns.

Study-note section FDB32

> SELECT Department, AVG(Salary) FROM Employee WHERE Department <> 'Sales' GROUP BY Department HAVING AVG(Salary) > 35000; lists qualifying department averages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 11-11

> Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 30 | lines 4-5

> • Verify relations and attributes exist. • Verify operations

Question sources

Addis Ababa University mock exam · Question 14

Question 14 | Q00219

Which of the following keys is generally used to represent the relationships between the tables?

A. None of the choices is correct

B. Secondary key

C. Primary key

D. Foreign key

Hint

Match the validity requirement to the appropriate constraint definition.

Answer D: Foreign key

A foreign key is the child-side reference linking to an existing parent candidate key.

Why each choice is right or wrong

A - Incorrect: A named key does supply the link (PDF 48.)

B - Incorrect: Secondary key is not the child reference (PDF 48.)

C - Incorrect: Primary key is the target identifier (PDF 48.)

D - Correct: Foreign key is the explicit link (PDF 48.)

Primary note proof

Study-note section FDB15

Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Study-note section FDB07

A primary key is one selected candidate key; remaining candidates are alternate keys.

Study-note section FDB25

Secondary indexes provide other access paths and must represent all searchable values with record pointers or pointer lists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 48 | lines 14-15

- A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key

Question sources

Mock exam 4 · Question 21

Question 15 | Q00220

Which of the following refers to the level of data abstraction that describes exactly how the data is actually stored?

A. Conceptual Level

B. Physical Level

C. File Level

D. Logical Level

Hint

Trace which schema levels a proposed change affects.

Answer B: Physical Level

Physical abstraction describes storage-media details.

Why each choice is right or wrong

A - Incorrect: Conceptual hides storage (PDF 7.)

B - Correct: Physical covers storage details (PDF 7.)

C - Incorrect: File is not a three-schema level (PDF 7.)

D - Incorrect: Logical describes structure rather than layout (PDF 7.)

Primary note proof

Study-note section FDB04

> The external level gives each user group a selected view. The conceptual level describes the whole logical database: entity types, relationships, data types and constraints. The internal/physical level describes files, record placement and access paths. These are abstraction levels, distinct from client/server deployment tiers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 12-13

> 2. Low-level or physical data models provide concepts that describe the details of how data is stored on the computer storage media. Concepts provided by physical data models are generally meant

Question sources

Mock exam 4 · Question 22

Question 16 | Q00221

Which one of the following given statements possibly contains the error?

A. select empid where empid = 1009 and Lastname = 'ADDIS';

B. select empid from emp where empid = 10006;

C. select empid from emp;

D. select * from emp where empid = 10003;

Hint

Trace how SELECT forms its result columns and rows.

Answer A: The first SELECT is missing its FROM source.

The first query reads empid and Lastname but never names a table in FROM. FDB27 allows literal-only SELECT statements without FROM, but this statement is not literal-only. The other three statements include emp as their source and use ordinary valid selection syntax, assuming those table and column names exist.

Why each choice is right or wrong

A - Correct: It refers to table columns without a FROM clause identifying their source.

B - Incorrect: It selects empid from emp with a syntactically ordinary equality predicate.

C - Incorrect: It supplies emp as the table source and validly selects its empid column.

D - Incorrect: It supplies a table source, selects all columns, and filters by empid.

Primary note proof

Study-note section FDB27

> SELECT Name,Address FROM Customer; produces only those mailing-label fields. SELECT * selects all columns; DISTINCT removes duplicate result rows; AS supplies an alias; ALL permits duplicates and does not mean all columns. SQL normally preserves duplicates without DISTINCT. A query reading table columns needs an appropriate FROM source. A literal-only query such as SELECT 1 may legally omit FROM.

Study-note section FDB27

> SELECT alone normally reads without changing the database state.

Study-note section FDB27

> Typical written clause order is SELECT ... FROM ... WHERE ... GROUP BY ... HAVING ... ORDER BY ....

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 11-11

> Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Mock exam 4 · Question 23

Question 17 | Q00222

Which one of the following is a type of Data Manipulation Command?

A. Delete

B. Alter

C. All choices are correct

D. Create

Hint

Classify each SQL command by the kind of change it performs.

Answer A: Delete

DELETE is explicitly DML; ALTER and CREATE are DDL.

Why each choice is right or wrong

A - Correct: DELETE is in DML list (PDF 26, 26.)

B - Incorrect: ALTER is in DDL list (PDF 26, 26.)

C - Incorrect: Other choices include DDL (PDF 26, 26.)

D - Incorrect: CREATE is in DDL list (PDF 26, 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 8-9

> Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME

Fundamental of Database Systems module | PDF page 26 | lines 11-11

> Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Mock exam 4 · Question 24

Question 18 | Q00223

Which one of the following refers to the "data about data"?

A. Warehouse

B. Directory

C. Sub Data

D. Meta Data

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer D: Meta Data

The description of database structure and constraints is metadata.

Why each choice is right or wrong

A - Incorrect: Warehouse is not this term (PDF 5.)

B - Incorrect: Directory is not this term (PDF 5.)

C - Incorrect: Sub Data is not this term (PDF 5.)

D - Correct: Meta-data describes the data structure (PDF 5.)

Primary note proof

Study-note section FDB03

> A schema describes names, structures and constraints; an instance, state or snapshot is the data at a particular time. Inserting a student changes the state, while adding a column changes the schema. Metadata is data describing data. The catalog or data dictionary records table definitions, types, constraints and often access privileges and optimizer statistics. A schema diagram draws tables and key references; labels such as database diagram can overlap.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 28-29

> description of the database structure and constraints. This information is called meta-data, and it describes the structure of the database.

Question sources

Mock exam 4 · Question 25

Question 19 | Q00298

A language that used to formulate the schema of database structure is_____.

A. Data Manipulation language

B. Data Definition Language

C. Transaction Control Language

D. Data Control Language

Hint

Classify each SQL command by the kind of change it performs.

Answer B: Data Definition Language

DDL supplies definitions, while DML handles data operations.

Why each choice is right or wrong

A - Incorrect: Manipulates contents (PDF 26.)

B - Correct: Defines database structure (PDF 26.)

C - Incorrect: Controls transactions (PDF 26.)

D - Incorrect: Controls privileges (PDF 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 5-6

> has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language).

Question sources

Computer Science model exam · Question 29

University of Gondar model exam · Question 29

Question 20 | Q00299

A table can have only one

A. Unique Key

B. Candidate key

C. Primary key

D. Foreign key

Hint

Test uniqueness and minimality separately when classifying keys.

Answer C: Primary key

One candidate is selected as the primary key; multiple candidates and multiple inter-table references can coexist.

Why each choice is right or wrong

A - Incorrect: Uniqueness need not be the selected identifier (PDF 11, 11.)

B - Incorrect: Several candidates are explicitly allowed (PDF 11, 11.)

C - Correct: Primary key is the selected candidate (PDF 11, 11.)

D - Incorrect: Separate relationships can use separate foreign keys (PDF 11, 11.)

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

> A table can have several candidates and foreign keys but at most one declared primary-key constraint, possibly containing several columns.

Study-note section FDB07

> Prefer stable, compact identifiers when practical, but Email, SSN or EmployeeID is a candidate only if business rules guarantee uniqueness.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 10-11

> Primary Key – A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Fundamental of Database Systems module | PDF page 11 | lines 9-10

> An entity set may have more than one candidate key. Primary Key – A primary key is one of the candidate keys chosen by the database designer to

Question sources

Computer Science model exam · Question 30

University of Gondar model exam · Question 30

Question 21 | Q00301

When the values in one or more attributes being used as a foreign key must exist there must be a corresponding primary key value in another table in the database, we have

created

A. Domain Integrity constraint

B. Referential Integrity Constraint

C. Transitive Dependency

D. Entity Integrity Constraints

Hint

Match the validity requirement to the appropriate constraint definition.

Answer B: Referential Integrity Constraint

Matching foreign-key values to existing parent occurrences is the referential-integrity rule.

Why each choice is right or wrong

A - Incorrect: Domain checks legal attribute values (PDF 48.)

B - Correct: Matches the explicit parent-reference rule (PDF 48.)

C - Incorrect: Transitivity is functional dependency (PDF 48.)

D - Incorrect: Entity integrity constrains the primary key itself (PDF 48.)

Primary note proof

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Study-note section FDB17

> Full dependency requires all determinant attributes; partial dependency holds through a proper subset.

Study-note section FDB17

A prime attribute belongs to some candidate key; a nonprime attribute belongs to none.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 48 | lines 16-17

- Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Question sources

Computer Science model exam · Question 32

University of Gondar model exam · Question 32

Question 22 | Q00302

Which of the following is not a feature of DBMS?

A. Support ACID Property

B. Single-user Access only

C. High Level of Security

D. Minimum Duplication and Redundancy of Data

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer B: Single-user Access only

Single-user-only contradicts the explicit multiuser DBMS requirement.

Why each choice is right or wrong

A - Incorrect: ACID is a requirement (PDF 6, 66.)

B - Correct: Multiuser support disproves only (PDF 6, 66.)

C - Incorrect: Security is supported (PDF 6, 66.)

D - Incorrect: Normalization aims to reduce redundancy (PDF 6, 66.)

Primary note proof

Study-note section FDB01

A DBMS centralizes metadata and supports multiple views, authorization, transactions, backup and recovery. It controls redundancy rather than guaranteeing zero duplication.

Study-note section FDB37

A transaction groups operations into one logical unit. Atomicity applies all its effects or none. Consistency means a correct transaction preserves declared invariants. Isolation controls interference between concurrent transactions; serializable isolation provides behavior equivalent to some serial execution. Durability preserves committed effects after a crash, typically through durable logging rather than forcing every data page immediately.

Study-note section FDB01

Concurrent access itself is useful; uncontrolled concurrent access causes anomalies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 6 | lines 7-8

Sharing of data and multiuser transaction processing - A multiuser DBMS must allow multiple users to access the database at the same time. This is essential if data for multiple applications is

Fundamental of Database Systems module | PDF page 66 | lines 3-4

To ensure data integrity, DBMS should maintain the following ACID properties: • Atomicity: A transaction is an atomic unit of processing; it is either performed in its

Question sources

Computer Science model exam · Question 33

University of Gondar model exam · Question 33

Question 23 | Q00303

Which of the following is not a function of the database?

A. Analyzing code

B. Provide Backup and recovery

C. Manipulating data

D. Managing stored data

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer A: Analyzing code

Data definition, manipulation and recovery are database responsibilities. Arbitrary program-code analysis is not a listed database function; parsing database queries is a narrower task.

Why each choice is right or wrong

A - Correct: General code analysis is outside the stated functions (PDF 5, 97.)

B - Incorrect: Recovery is expressly covered (PDF 5, 97.)

C - Incorrect: Manipulation is expressly covered (PDF 5, 97.)

D - Incorrect: DBMS manages databases (PDF 5, 97.)

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB01

> A DBMS centralizes metadata and supports multiple views, authorization, transactions, backup and recovery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 15-16

> DBMS is a software that facilities the processes of defining, constructing, manipulating, and sharing databases. Example: MySQL server, MS ACESS, Microsoft SQL server.

Fundamental of Database Systems module | PDF page 97 | lines 4-5

> i. Purpose of Database Recovery ◦ To bring the database into the last consistent state, which existed prior to the

Question sources

Computer Science model exam · Question 34

University of Gondar model exam · Question 34

Question 24 | Q00307

Read each pair left→right. Buildings and locations are 1:1; each employee belongs to one department and a department may have many employees; a hospital serves many patients, each registered at one hospital; authors and books are M:N. Which pair is many-to-one?

A. [Building - Location]

B. [Employee - Department]

C. [Hospital - Patient]

D. [Author - Book]

Hint

Read the relationship in both directions and check participation constraints.

Answer B: B. [Employee - Department]

Many-to-one means several left-side entities may share a right-side entity while each left entity has at most one right partner. Employees share departments under the stated assignment rule. Building→Location is one-to-one, Hospital→Patient is one-to-many and Author→Book is many-to-many. Reading the direction and both maximum counts determines the answer rather than the entity names.

Why each choice is right or wrong

A - Incorrect: Building→Location is explicitly 1:1.

B - Correct: Many employees can share one department; each employee has one department.

C - Incorrect: Hospital→Patient is 1:N under the registration rule.

D - Incorrect: Author→Book is explicitly M:N.

Primary note proof

Study-note section FDB09

> Cardinality specifies maximum associated occurrences: 1:1, 1:N or M:N. Minimum cardinality specifies optional/partial participation (zero permitted) or mandatory/total participation (at least one). A maximum of one does not mean participation is compulsory. Relationship degree counts participating entity roles: unary recursive, binary or ternary.

Study-note section FDB09

> If each employee belongs to one department and each department has many employees, Employee→Department is N:1; Department→Employee is 1:N.

Study-note section FDB09

> Building→Location could also be N:1, so names alone cannot select a unique example.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 29-30

> One to many(1-M)- When entities in one entity set can take part only once in the relationship set and entities in other entity set can take part more than once in the relationship set, cardinality is

Question sources

Computer Science model exam · Question 38

University of Gondar model exam · Question 38

Question 25 | Q00374

_____ is a single elementary unit of information representing an attribute of an entity.

A. Entity Set

B. Record

C. File

D. Field

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer D: Field.

A field corresponds to one attribute rather than the collection of attributes describing a whole record. FDB03 equates attribute, column and field, while tuple, row and record refer to related attribute values together. An entity set groups entities, and a file groups stored records, so both are larger collections.

Why each choice is right or wrong

A - Incorrect: An entity set collects entities of one type rather than representing a single attribute.

B - Incorrect: A record contains related values for multiple attributes, rather than one elementary field.

C - Incorrect: A file groups records and is not an elementary attribute unit.

D - Correct: A field represents one attribute or property within a record.

Primary note proof

Study-note section FDB03

> A relation is a set of tuples, represented as a table. A tuple/row/record contains related attribute values; an attribute/column/field describes a property. Degree counts columns; cardinality counts rows. Classical relations have distinct tuples, atomic cells and domains for attributes; row order has no meaning. Column order has no logical meaning when attributes are identified by name. SQL tables can contain duplicate rows unless constrained.

Study-note section FDB06

> An entity type describes common properties; its current collection of entities is an entity set.

Study-note section FDB06

> In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 13 | lines 25-26

entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.

Question sources

Data Structures & Algorithms practice questions · Question 4

Question 26 | Q00548

An ER model has entity types A and B connected by an attribute-free relationship R. Cardinalities are read A→B. Which conditions allow R to be represented by a single B-key foreign-key column in A, permitting NULL for partial participation? Select all that apply.

A. Relationship R is one -to -many and the participation of A in R is Total.

B. Relationship R is one -to -many and the participation of A in R is Partial.

C. Relationship R is many -to -one and the participation of A in R is Total.

D. Relationship R is many --to -one and the participation of A in R is Partial.

Hint

Locate the relationship's cardinality and participation before placing a foreign key.

Answer C, D: C. Relationship R is many -to -one and the participation of A in R is Total.; D. Relationship R is many --to -one and the participation of A in R is Partial.

A single foreign-key column in A can represent R whenever each A is related to at most one B. This is exactly the many-to-one direction. Total participation makes the reference mandatory; partial participation permits NULL. One-to-many in the opposite direction needs the reference on B or another representation, because one A may have multiple related B values.

Why each choice is right or wrong

A - Incorrect: For A→B one-to-many, B normally holds A's key; total participation does not reverse that mapping.

B - Incorrect: Partial participation does not make a one-to-many collection fit a single ordinary foreign key in A.

C - Correct: Many-to-one with total A participation stores a non-NULL B reference in A.

D - Correct: Many-to-one with partial A participation stores a nullable B reference in A.

Primary note proof

Study-note section FDB11

For 1:N, place the one-side key as a foreign key in the many-side table; store relationship attributes there. Department(DeptID) and Employee(EmpID,DeptID,HireDate) implement many employees per department. If participation is total, make the reference NOT NULL. If partial, a nullable reference can represent no assignment.

Study-note section FDB11

For A→B many-to-one, A can absorb the relationship reference under either total or partial participation; requiring total participation as the only possible case overstates the rule.

Study-note section FDB11

Prefer the totally participating side to reduce NULLs; NOT NULL enforces that side's mandatory reference.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 12 | lines 18-19

4. For each binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type. Include the primary key of the relation T as

Question sources

Computer Science model exam · 2015 · Question 17

Question 27 | Q00567

Under set relational algebra, which listed operation is not commutative? Interpret Join as an inner join, adjust its predicate when swapping inputs, and ignore attribute order.

A. Union

B. Join

C. Intersection

D. None

Hint

Apply each algebra operator to the input relations step by step.

Answer D: D. None

Union and intersection give the same tuple sets when their inputs are exchanged. An inner join also gives corresponding matches after the predicate is adjusted for the swapped inputs; the question explicitly ignores attribute order. Thus none of the three named operations lacks commutativity here. This conclusion depends on the stated inner-join interpretation; preservation of one designated outer side would use a different rule.

Why each choice is right or wrong

A - Incorrect: Union is commutative on union-compatible relations.

B - Incorrect: Inner join is commutative under the stated predicate and schema convention.

C - Incorrect: Intersection is commutative on union-compatible relations.

D - Correct: All three named operators are commutative under these assumptions.

Primary note proof

Study-note section FDB35

> Selection σ filters tuples; projection π retains attributes and removes duplicate tuples under set semantics. Rename ρ changes relation or attribute names. Union, difference and intersection require union-compatible schemas. Difference R−S is not commutative; union and intersection are. Inner joins are commutative up to attribute order when predicates are adjusted; left outer joins are not generally commutative. Identify the join variant before predicting its behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 15-16

> Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Fundamental of Database Systems module | PDF page 24 | lines 16-17

> Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples). 6.2 Relational Calculus

Question sources

Computer Science model exam · 2015 · Question 36

Question 28 | Q00595

Which of the following is a benefit of normalization rather than a potential cost or disadvantage?

A. May require joins to reconstruct decomposed records

B. May reduce system performance

C. Time consuming

D. Reduce duplicate data

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer D: Reduce duplicate data

Normalization organizes facts according to their dependencies to reduce avoidable duplication and update anomalies. That reduction is a benefit. Decomposing a table may require joins when an application needs the original combined information, which can increase retrieval cost for some workloads. Dependency analysis and schema design also take time. These are possible costs, not claims that normalization always slows a system. Declared dependencies suffice for normalization even when the current sample data shows no errors.

Why each choice is right or wrong

A - Incorrect: A possible cost: reconstructing information from decomposed tables may require joins.

B - Incorrect: A possible cost: additional joins can reduce performance for some workloads, although this is not inevitable.

C - Incorrect: A possible cost: analyzing dependencies and designing a decomposition takes time.

D - Correct: Correct: reducing duplicate facts is a principal benefit, rather than a disadvantage.

Primary note proof

Study-note section FDB19

> Normalization separates facts according to their determinants, reducing avoidable repetition and associated anomalies. It does not require seeing bad current data: declared dependencies suffice. Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design. Reduced redundancy can improve integrity and storage use, but extra keys and joins mean neither storage savings nor faster retrieval is guaranteed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 18-19

> Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. It is a technique of organizing the data in the

Question sources

Computer Science model exam · 2015 · Question 64

Question 29 | Q00617

Let you have two relations A and B, each table has 200 and 100 rows of records respectively. So after applying cross product on those tables, how many rows of records you will have?

A. 300

B. 20000

C. 2000

D. 100

Hint

Apply each algebra operator to the input relations step by step.

Answer B: 20000

The Cartesian-product example explicitly creates 1000*50 tuples from inputs of sizes 1000 and 50 (PDF p.28, lines 6-8). Applying the same all-pairs construction to 200 and 100 rows yields 200*100 = 20000. Each of the 200 rows is paired with each of the 100 rows; no filtering or duplicate-eliminating projection is specified.

Why each choice is right or wrong

A - Incorrect: 300 adds input sizes instead of counting pairs (PDF p.28).

B - Correct: 20000 is 200*100, the illustrated Cartesian pair-count rule (PDF p.28).

C - Incorrect: 2000 is ten times too small for the all-pairs count (PDF p.28).

D - Incorrect: 100 counts only one input and omits its 200 pairings per row (PDF p.28).

Primary note proof

Study-note section FDB35

> A Cartesian product has |R|×|S| tuples and degree degree(R)+degree(S), renaming conflicting attributes if necessary. Thus 200×100=20,000 and 5×7=35. Join equals selection over a product; intersection can be derived from difference. A common minimal basis contains selection, projection, union, difference, product and rename. Assignment is a useful notation, not an indispensable primitive; lists of basic operations vary across textbooks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 11-14

> Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2.

Fundamental of Database Systems module | PDF page 28 | lines 6-8

> ◦ Requires (1000+50) disk accesses to read from Staff and Branch relations ◦ Creates temporary relation of Cartesian Product (1000*50) tuples ◦ Requires (1000*50) disk access to read in temporary relation and test predicate

Question sources

Computer Science model exam · 2015 · Question 86

Question 30 | Q00675

_____It defines how and where data are organized in physical data storage

A. Internal Schema

B. Conceptual Schema

C. External Schema

D. External and Conceptual Schema

Hint

Trace which schema levels a proposed change affects.

Answer A: Internal Schema

Internal schema specifies physical structure and access paths.

Why each choice is right or wrong

A - Correct: Exact storage schema (PDF 7.)

B - Incorrect: Conceptual hides physical detail (PDF 7.)

C - Incorrect: External is user view (PDF 7.)

D - Incorrect: Neither listed user-level schema defines storage (PDF 7.)

Primary note proof

Study-note section FDB04

> The external level gives each user group a selected view. The conceptual level describes the whole logical database: entity types, relationships, data types and constraints. The internal/physical level describes files, record placement and access paths. These are abstraction levels, distinct from client/server deployment tiers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 28-29

> 1. The internal level has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of

Question sources

Computer Science model exam 2 · Question 16

Question 31 | Q00676

Which database-design stage primarily builds an ER model from the users' requirements?

A. Applications and security

B. Physical refinement

C. Conceptual database

D. Schema refinement

Hint

Identify the design phase from its inputs and intended deliverable.

Answer C: Conceptual database

Conceptual design represents the required entities, attributes and relationships in an ER model. Logical design later maps that conceptual model into relations, keys and constraints and checks normalization. Physical refinement chooses storage organizations and access paths, while application and security work concerns behavior and protection rather than the modeling stage requested. The stem asks for the primary stage building the ER model, not every later activity that might consult it.

Why each choice is right or wrong

A - Incorrect: Incorrect: application and security work addresses behavior and protection, rather than building the conceptual ER model.

B - Incorrect: Incorrect: physical refinement chooses storage organization and access paths.

C - Correct: Correct: conceptual design turns requirements into the ER model.

D - Incorrect: Incorrect: schema refinement checks or improves a logical schema, rather than naming the initial ER-model stage.

Primary note proof

Study-note section FDB05

> First collect business facts, required transactions, reports, volumes, privacy rules and quality targets. Conceptual design turns them into an ER/EER model. Logical design maps the conceptual model to relations, keys and constraints and checks normalization. Physical design chooses file organization, indexes and storage. Implementation creates and loads the actual objects; testing checks requirements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 4-5

> High-level conceptual data models provide concepts for presenting data in ways that are close to the way people perceive data. A typical example is the entity relationship model, which uses main

Question sources

Computer Science model exam 2 · Question 17

Question 32 | Q00678

Employee(empid, ...) already exists, has no primary key, and empid values are unique and non-NULL. Which SQL statement adds a primary key on empid?

A. ADD TABLE Employee ADD PRIMARY KEY (empid)

B. ATTACH TABLE Employee ADD PRIMARY KEY (empid)

C. ADD TABLE Employee ATTACH PRIMARY KEY (empid)

D. ALTER TABLE Employee ADD PRIMARY KEY (empid);

Hint

Distinguish table definition changes from changes to stored rows.

Answer D: D. ALTER TABLE Employee ADD PRIMARY KEY (empid);

Adding a constraint to an existing table uses ALTER TABLE, followed by ADD PRIMARY KEY and the parenthesized key column. The existing unique, non-NULL empid values satisfy the prerequisite for the constraint. Option D has this structure. ADD TABLE and ATTACH TABLE do not perform this alteration, and ATTACH PRIMARY KEY is not the required constraint clause.

Why each choice is right or wrong

A - Incorrect: ADD TABLE does not modify an existing table's constraints.

B - Incorrect: ATTACH TABLE is not the required table-alteration syntax.

C - Incorrect: Neither ADD TABLE nor ATTACH PRIMARY KEY forms this SQL alteration.

D - Correct: The corrected ALTER TABLE statement adds the primary-key constraint.

Primary note proof

Study-note section FDB30

> CREATE TABLE Employee (EmpID INTEGER PRIMARY KEY, Name VARCHAR(80)); declares two columns. The name and its type stay together; the comma separates whole column declarations. To add a primary key later: ALTER TABLE Employee ADD PRIMARY KEY (EmpID); existing data must be unique and non-NULL. A table cannot gain a second primary key without replacing the existing constraint.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 8-9

> Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME

Question sources

Computer Science model exam 2 · Question 19

Question 33 | Q00679

While checking normalization process, the problem that doesn't exist on transitive dependency is

A. Modification Anomaly

B. Revoke Anomaly

C. Insertion Anomaly

D. Deletion Anomaly

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer B: Revoke Anomaly

The defined anomalies are insertion, deletion and update; revoke concerns privileges.

Why each choice is right or wrong

A - Incorrect: Modification means update (PDF 16.)

B - Correct: Revoke is not a normalization anomaly (PDF 16.)

C - Incorrect: Insertion is defined (PDF 16.)

D - Incorrect: Deletion is defined (PDF 16.)

Primary note proof

Study-note section FDB19

> Suppose Enrollment repeats StudentName for every course. An update anomaly requires changing every copy of the student's name; missed copies disagree. An insertion anomaly prevents recording a new student until some enrollment exists. A deletion anomaly loses the only record of a student's details when their last enrollment is deleted. Revoke anomaly is not a normalization anomaly.

Study-note section FDB19

> Normalization separates facts according to their determinants, reducing avoidable repetition and associated anomalies.

Study-note section FDB19

> Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 8-9

data. Types of database anomalies are Insertion anomaly, Deletion anomaly, and Update anomaly. Insertion Anomaly - occurs when the entire primary key is not known and the database cannot

Question sources

Computer Science model exam 2 · Question 20

Question 34 | Q00680

You need to produce a report for mailing labels for customers. The mailing label must have only the customer name and address. The CUSTOMERS table has these columns, ID ,NAME,ADDRESS,PHONE ,the SELECT statement to accomplishes this task?
*

A. SELECT name, address FROM customers;

B. SELECT*FROM customers;

C. SELECT id, name, address, phone FROM customers;

D. SELECT cust_name, cust_address FROM customers;

Hint

Trace how SELECT forms its result columns and rows.

Answer A: SELECT name, address FROM customers;

The required output contains exactly NAME and ADDRESS from the listed schema. FDB27 distinguishes naming specific columns from selecting all columns with *. A names the two available columns explicitly. B and C include unwanted ID and PHONE fields, while D uses column names not supplied by the question.

Why each choice is right or wrong

A - Correct: It selects exactly the two existing columns required for the mailing label.

B - Incorrect: The wildcard also includes ID and PHONE, exceeding the required output.

C - Incorrect: It explicitly includes ID and PHONE in addition to the requested label fields.

D - Incorrect: The supplied schema names NAME and ADDRESS, not CUST_NAME and CUST_ADDRESS.

Primary note proof

Study-note section FDB27

SELECT Name,Address FROM Customer; produces only those mailing-label fields. SELECT * selects all columns; DISTINCT removes duplicate result rows; AS supplies an alias; ALL permits duplicates and does not mean all columns. SQL normally preserves duplicates without DISTINCT. A query reading table columns needs an appropriate FROM source. A literal-only query such as SELECT 1 may legally omit FROM.

Study-note section FDB27

> Typical written clause order is SELECT ... FROM ... WHERE ... GROUP BY ... HAVING ... ORDER BY ....

Study-note section FDB27

> SELECT alone normally reads without changing the database state.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 12-13

> Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations.

Fundamental of Database Systems module | PDF page 26 | lines 11-11

> Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Computer Science model exam 2 · Question 21

Question 35 | Q00681

If the maximum cardinality is equal to 1 in both directions of a relationship, then it is classified as:

A. Optional

B. Many-to-many

C. One-to-one

D. One-to-many

Hint

Read the relationship in both directions and check participation constraints.

Answer C: One-to-one

At most one related entity on each side is the one-to-one case.

Why each choice is right or wrong

A - Incorrect: Optional describes minimum participation (PDF 11.)

B - Incorrect: Many-to-many allows multiple (PDF 11.)

C - Correct: Both maxima match one-to-one (PDF 11.)

D - Incorrect: One-to-many permits multiple on one side (PDF 11.)

Primary note proof

Study-note section FDB09

> Cardinality specifies maximum associated occurrences: 1:1, 1:N or M:N. Minimum cardinality specifies optional/partial participation (zero permitted) or mandatory/total participation (at least one). A maximum of one does not mean participation is compulsory. Relationship degree counts participating entity roles: unary recursive, binary or ternary.

Study-note section FDB09

> Read business rules in both directions.

Study-note section FDB09

> If each employee belongs to one department and each department has many employees, Employee→Department is N:1; Department→Employee is 1:N.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 27-28

> One to one (1-1) - When each entity in each entity set can take part only once in the relationship, the cardinality is one to one.

Question sources

Computer Science model exam 2 · Question 22

Question 36 | Q00682

Which SQL statement sets PRICE to 600 in BOOK for rows whose ID is 'B001'?

A. CHANGE BOOK SET PRICE =600 Where ID ='B001';

B. UPDATE TO CHANGE BOOK SET PRICE =600 Where ID ='B001';

C. UPDATE BOOK SET PRICE =600 Where ID ='B001';

D. GRANT BOOK SET PRICE =600 Where ID ='B001';

Hint

Track which rows each data-changing command affects.

Answer C: UPDATE BOOK SET PRICE =600 Where ID ='B001';

UPDATE changes existing row values. The table name follows UPDATE, SET supplies the assignment and WHERE restricts the rows. C therefore assigns 600 only where ID equals the quoted string B001. CHANGE is not the row-update command, the extra TO CHANGE words break the required UPDATE structure, and GRANT controls privileges rather than performing a row assignment. These malformed alternatives are definite incorrect selections.

Why each choice is right or wrong

A - Incorrect: Incorrect: the required row-change statement begins UPDATE, not CHANGE BOOK SET.

B - Incorrect: Incorrect: UPDATE must be followed by its target table, not TO CHANGE.

C - Correct: Correct: the statement assigns 600 to PRICE for precisely the matching ID rows.

D - Incorrect: Incorrect: GRANT manages privileges; it cannot perform this SET assignment.

Primary note proof

Study-note section FDB29

> UPDATE Book SET Price = 600 WHERE ID = 'B001'; changes matching rows. UPDATE without WHERE targets every row. DELETE FROM Publisher WHERE Name = 'XWZ'; deletes matching publisher rows, subject to reference policies. DELETE FROM r; removes every row but leaves r defined. Both clear relation entries and delete rows describe that effect. Several UPDATE statements can all be valid while performing different tasks; most appropriate requires a requested change and a specified quoting dialect.

Study-note section FDB26

> DCL manages privileges through GRANT and REVOKE.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 5-6

> has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language).

Question sources

Computer Science model exam 2 · Question 23

Question 37 | Q00683

While listing limitation of File Processing System identify the wrong statement?

A. Difficult in accessing data

B. Data isolation and integrity.

C. Data redundancy & inconsistency available.

D. Concurrent access is possible.

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer D: Concurrent access is possible.

The file-system limitation is concurrent-access anomalies; merely allowing concurrent access is not itself the listed limitation.

Why each choice is right or wrong

A - Incorrect: Access difficulty is listed (PDF 5.)

B - Incorrect: Isolation and integrity problems are listed (PDF 5.)

C - Incorrect: Redundancy and inconsistency are listed (PDF 5.)

D - Correct: Possibility of access does not name the anomaly (PDF 5.)

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB01

> Concurrent access itself is useful; uncontrolled concurrent access causes anomalies.

Study-note section FDB01

> Traditional file applications define their own formats and access procedures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 21-22

> Integrity problems, Atomicity problems, Concurrent access anomalies. 1.3 Characteristics of the Database Approach

Question sources

Computer Science model exam 2 · Question 24

Question 38 | Q00684

The Entity Relationship (E-R) Model

A. Entities are represented by circle

B. Attributes are represented by rectangle

C. It represents conceptual view

D. It corresponds to row

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer C: It represents conceptual view

ER depicts the conceptual database schema and entity rectangles.

Why each choice is right or wrong

A - Incorrect: Entity uses rectangle (PDF 9, 10.)

B - Incorrect: Attribute is a property rather than an entity rectangle (PDF 9, 10.)

C - Correct: Conceptual depiction matches ER (PDF 9, 10.)

D - Incorrect: A model is not one tuple (PDF 9, 10.)

Primary note proof

Study-note section FDB06

> An entity is a distinguishable real or conceptual thing, such as one student or one course. An entity type describes common properties; its current collection of entities is an entity set. A relationship associates entities. In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes. Always inspect the diagram legend.

Study-note section FDB05

> Conceptual design turns them into an ER/EER model.

Study-note section FDB05

> Logical design maps the conceptual model to relations, keys and constraints and checks normalization.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 28-29

> ERD a graphical technique for representing a database schema. It is a modeling tool used to depict graphically a DB design before it is actually implemented. It shows the various entities being

Fundamental of Database Systems module | PDF page 10 | lines 5-6

> or a university course). In ER diagrams, an entity is placed in a rectangle. Each entity has attributes, the particular properties that describe it. Entities that do not have key

Question sources

Computer Science model exam 2 · Question 25

Question 39 | Q00685

Removal of transitivity property should be full filled in

A. First Normal form

B. Fourth Normal form

C. Third Normal form

D. Second Normal form

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer C: Third Normal form

3NF requires nonprime attributes to depend nontransitively on keys after reaching 2NF.

Why each choice is right or wrong

A - Incorrect: 1NF requires atomicity (PDF 18.)

B - Incorrect: 4NF is not the stated rule (PDF 18.)

C - Correct: 3NF explicitly removes transitive dependence (PDF 18.)

D - Incorrect: 2NF removes partial dependence (PDF 18.)

Primary note proof

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key. For Enrollment(SID,CID,SName,CTitle,Grade), assume key (SID,CID), SID→SName and CID→CTitle. Split into Student(SID,SName), Course(CID,CTitle) and Enrollment(SID,CID,Grade). Grade still depends on the whole pair. A relation whose every candidate key is single-attribute has no nonprime partial-key dependency. Higher forms inherit 2NF, so no partial dependencies establishes the minimum guaranteed stage only when 1NF is also assumed.

Study-note section FDB21

> With atomic values the highest form is 1NF, even though a transitive dependency also exists.

Study-note section FDB21

> The mnemonic whole key and nothing but the key describes the common single-key case; it cannot replace the full tests when keys overlap.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 18 | lines 7-8

> _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Computer Science model exam 2 · Question 26

Question 40 | Q00686

If relation R has 5 tuples and relation S has also 7 tuples, then Cartesian product of R and S will have

A. 5+7 tuples

B. 7-5 tuples

C. 5X7 tuples

D. 5/7 tuples

Hint

Apply each algebra operator to the input relations step by step.

Answer C: 5X7 tuples

The module explicitly multiplies input tuple counts for a Cartesian product (PDF p.28, lines 6-8). Hence 5 rows paired with 7 rows give 5*7 = 35 distinct concatenated tuples, before any filtering/projection.

Why each choice is right or wrong

A - Incorrect: 5+7 is the sum, not all pair combinations (PDF p.28).

B - Incorrect: 7-5 is the size difference, not the pair count (PDF p.28).

C - Correct: 5*7 gives the 35 combinations required by the product construction (PDF p.28).

D - Incorrect: 5/7 is not even an integer tuple count and is not pair enumeration (PDF p.28).

Primary note proof

Study-note section FDB35

> A Cartesian product has |R|×|S| tuples and degree degree(R)+degree(S), renaming conflicting attributes if necessary. Thus 200×100=20,000 and 5×7=35. Join equals selection over a product; intersection can be derived from difference. A common minimal basis contains selection, projection, union, difference, product and rename. Assignment is a useful notation, not an indispensable primitive; lists of basic operations vary across textbooks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 11-14

> Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2.

Fundamental of Database Systems module | PDF page 28 | lines 6-8

◦ Requires (1000+50) disk accesses to read from Staff and Branch relations ◦ Creates temporary relation of Cartesian Product (1000*50) tuples ◦ Requires (1000*50) disk access to read in temporary relation and test predicate

Question sources

Computer Science model exam 2 · Question 27

Question 41 | Q00688

The language associated with a database management system that is employed by end users and programmers to alter data values in the database is the:

A. Data translation language.

B. Data definition language.

C. Data manipulation language.

D. Data presentation language.

Hint

Classify each SQL command by the kind of change it performs.

Answer C: Data manipulation language.

The language for INSERT, UPDATE and DELETE is DML.

Why each choice is right or wrong

A - Incorrect: Not a defined language category (PDF 26.)

B - Incorrect: DDL defines structure (PDF 26.)

C - Correct: DML changes stored data (PDF 26.)

D - Incorrect: Not a defined language category (PDF 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 6-7

> language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language)

Question sources

Computer Science model exam 2 · Question 29

Question 42 | Q00730

Using standard SQL string literals, which update changes only the name of the student whose ID is '123' to 'abel'?

A. UPDATE student ID='123';

B. UPDATE student SET ID='123';

C. UPDATE student SET name='abel' WHERE ID='123';

D. UPDATE student SET name='abel';

Hint

Track which rows each data-changing command affects.

Answer C: C. UPDATE student SET name='abel' WHERE ID='123';

UPDATE needs SET to state the assignment and WHERE to restrict the target rows. C assigns the desired name only to the student with ID '123'. B changes IDs in every row, while D changes every student's name because it has no filter. A omits SET and is malformed. Correct syntax and the requested scope both matter.

Why each choice is right or wrong

A - Incorrect: SET is missing before the assignment.

B - Incorrect: This sets every student's ID and does not change the name.

C - Correct: SET changes name and WHERE restricts the operation to ID '123'.

D - Incorrect: Without WHERE, every name is changed.

Primary note proof

Study-note section FDB29

> INSERT INTO Hotel(HotelID,HotelName,SubCity) VALUES (1,'Hilton','Yeka'); pairs each named column with a compatible value. Counts and types must match; keys and other constraints still apply. Without a column list, values must follow the table's declared column order.

Study-note section FDB29

> UPDATE Book SET Price = 600 WHERE ID = 'B001'; changes matching rows.

Study-note section FDB28

> Use single quotes for SQL string literals: WHERE ID = 'B001'.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 5-6

> has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language).

Question sources

Computer Science model exam 2 · Question 71

Question 43 | Q00899

A weak entity is identified by its partial key together with the key of an independently keyed associated entity. Which listed terms describe that associated entity set? Select all that apply.

A. Identifying set

B. Strong entity set

C. Neighbor set

D. Owner set

Hint

Check how a dependent entity obtains its complete identity.

Answer B, D: B. Strong entity set; D. Owner set

A weak entity cannot identify itself using its own attributes alone. Its owner contributes the missing key component, and in the specified case that owner has an independent key, so it is a strong entity set. Both owner set and strong entity set describe it. The identifying relationship connects the sets; neighbor does not identify the required ownership role.

Why each choice is right or wrong

A - Incorrect: Identifying normally names the owner relationship rather than the defined owner entity-set role.

B - Correct: The owner supplying the independent key is normally a strong entity set.

C - Incorrect: Neighbor does not express the ownership and identification needed for a weak entity.

D - Correct: Owner set names the entity set whose key contributes to weak-entity identification.

Primary note proof

Study-note section FDB08

> A weak entity lacks a complete independent key. Its owner is normally a strong entity, so owner entity set and strong entity set may both describe the associated set. Its partial key, also called discriminator, distinguishes weak entities of the same owner. Identification combines the owner key and partial key.

Study-note section FDB08

> The weak entity participates totally in its identifying relationship.

Study-note section FDB08

> Chen notation uses a double rectangle, double identifying diamond and dashed underline for a partial key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 6-10

> Each entity has attributes, the particular properties that describe it. Entities that do not have key attributes of their own are called weak entity. The regular entity types that have a key attribute are called strong entity types. Weak entities are identified by being related to specific strong entities in combination with one of their attribute values. A weak entity type has a partial key, which is the attribute that can uniquely identify weak entities that are related to the same owner entity.

Fundamental of Database Systems module | PDF page 12 | lines 10-13

> 2. Choose one of the key attributes of E as the primary key for R. For each weak entity type W with owner entity type E (which is a strong entity), create a relation R & include all attributes of ‗W‘ as attributes of R. Also, include the primary key attribute of the relation that correspond to the owner entity type as foreign key attributes of R .

Question sources

Computer Science Exit Exam · 2018 · Question 12

Question 44 | Q00901

A language that allows the DBA or user to describe and name the entities, attributes, and relationships required for the application is

A. Transaction control language (TCL)

B. Data control language (DCL)

C. .
Data Manipulation Language (DML)

D. Data Definition Language (DDL)

Hint

Classify each SQL command by the kind of change it performs.

Answer D: Data Definition Language (DDL)

Defining schema objects belongs to DDL.

Why each choice is right or wrong

A - Incorrect: Transaction control is a different task (PDF 26.)

B - Incorrect: Privileges are a different task (PDF 26.)

C - Incorrect: Manipulation operates on contents (PDF 26.)

D - Correct: DDL supplies definitions (PDF 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 5-6

> has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language).

Question sources

Computer Science Exit Exam · 2018 · Question 14

Question 45 | Q00907

In relational-table terminology, which listed terms name a unit of related field values treated together by an application? Select all that apply.

A. Rows

B. Field

C. Record

D. Database

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer A, C: A. Rows; C. Record

A relational tuple groups related attribute values into one unit. Row and record are names for that same unit, so both options qualify even though one is plural. A field refers to one property rather than the whole grouping. A database is the larger coherent collection containing tables and their data, rather than one individual tuple.

Why each choice is right or wrong

A - Correct: A relational row groups related field values and is synonymous with a record.

B - Incorrect: A field represents one attribute, not the collection of related fields in the definition.

C - Correct: A record is the unit containing related fields described by the stem.

D - Incorrect: A database contains a larger coherent collection, rather than this single field grouping.

Primary note proof

Study-note section FDB03

> A relation is a set of tuples, represented as a table. A tuple/row/record contains related attribute values; an attribute/column/field describes a property. Degree counts columns; cardinality counts rows. Classical relations have distinct tuples, atomic cells and domains for attributes; row order has no meaning. Column order has no logical meaning when attributes are identified by name. SQL tables can contain duplicate rows unless constrained.

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application.

Study-note section FDB01

> It controls redundancy rather than guaranteeing zero duplication.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 13 | lines 25-26

entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.

Question sources

Computer Science Exit Exam · 2018 · Question 20

Question 46 | Q00927

Which template correctly creates a table with one column? Treat table_name, column_name and column_type as placeholders for valid identifiers and a data type.

A. CREATE table_name (column_name, column type);

B. CREATE table_name (column_type, column name);

C. CREATE TABLE table_name (column_name column_type);

D. CREATE TABLE table_name (column_type, column_name);

Hint

Distinguish table definition changes from changes to stored rows.

Answer C: C. CREATE TABLE table_name (column_name column_type);

A column declaration places its name immediately before its data type. Commas separate complete declarations when more than one column is present. The corrected C uses CREATE TABLE and a properly paired name and type. A and B omit TABLE, while D separates and reverses the name/type parts rather than declaring the intended column.

Why each choice is right or wrong

A - Incorrect: TABLE is missing and the name/type pair is split by a comma.

B - Incorrect: TABLE is missing and the type/name order is wrong.

C - Correct: CREATE TABLE is followed by one correctly ordered name/type declaration.

D - Incorrect: A comma separates type from name and no valid intended declaration is formed.

Primary note proof

Study-note section FDB30

> CREATE TABLE Employee (EmpID INTEGER PRIMARY KEY, Name VARCHAR(80)); declares two columns. The name and its type stay together; the comma separates whole column declarations. To add a primary key later: ALTER TABLE Employee ADD PRIMARY KEY (EmpID); existing data must be unique and non-NULL. A table cannot gain a second primary key without replacing the existing constraint.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 49 | lines 19-20

> - CREATE TABLE table-name (attr2 attr-type NOT NULL, forbids NULL values

Question sources

Computer Science Exit Exam · 2018 · Question 40

Question 47 | Q00932

From the following alternatives which one is FALSE about database system?

A. " I is very difficult to protect a file under the system

B. * iy database system the user is not required to write the procedures

C. « tn database system, DBMS provides a good protection mechanism.

D. ° i contains a wide variety of sophisticated techniques to store and retrieve the data.

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer A: " I is very difficult to protect a file under the system

DBMS permission and authentication support contradicts the claim of poor protection; B is interpreted as nonprocedural querying rather than no application code ever.

Why each choice is right or wrong

A - Correct: Contradicts described protection (PDF 53.)

B - Incorrect: Nonprocedural retrieval is supported (PDF 53.)

C - Incorrect: Matches security support (PDF 53.)

D - Incorrect: Storage and indexes support retrieval (PDF 53.)

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB25

> B/B+ trees support ordered and range search; hashing targets equality; bitmap indexes suit some analytic workloads.

Study-note section FDB01

> Complexity, administration, storage and licensing can outweigh benefits for a tiny fixed application; lightweight embedded DBMSs also exist.

Related reading (context, not primary note proof)

Study-note section FDB36

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 53 | lines 15-16

- DBMS can provide some security ◦ Each user has an account, username and password

Question sources

Computer Science Exit Exam · 2018 · Question 45

Question 48 | Q00943

Consider a directed line(->) from the relationship set advisor to both entity sets instructor and student. This indicates cardinality

A. Many to one

B. Many to many

C. ¢.
One to one

D. One to many

Hint

Read the relationship in both directions and check participation constraints.

Answer C: One-to-one, under the common relationship-arrow convention.

FDB09 explicitly states the convention used for this example: arrows from advisor toward both Instructor and Student impose maximum-one at both ends. Each instructor can therefore be associated with at most one student and each student with at most one instructor. That is 1:1, without implying mandatory participation. A different notation would need an explicit legend.

Why each choice is right or wrong

A - Incorrect: Many-to-one constrains only one end to maximum one, whereas both ends have arrows here.

B - Incorrect: Many-to-many permits multiple associations at both ends, contrary to both maximum-one arrows.

C - Correct: Maximum one at each end yields a one-to-one relationship under the stated common convention.

D - Incorrect: One-to-many allows multiple associations at one end, which the second arrow rules out.

Primary note proof

Study-note section FDB09

Read business rules in both directions. If each employee belongs to one department and each department has many employees, Employee→Department is N:1; Department→Employee is 1:N. Building→Location could also be N:1, so names alone cannot select a unique example. Under the common arrow convention, arrows from advisor to both Instructor and Student impose maximum-one at both ends, giving 1:1. Without a notation convention, arrows are unsafe evidence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 27-28

> One to one (1-1) - When each entity in each entity set can take part only once in the relationship, the cardinality is one to one.

Question sources

Computer Science Exit Exam · 2018 · Question 56

Question 49 | Q00957

What is the basic goal of Normalization?

A. To decrease data integrity

B. To reduce redundancy

C. ¢.
To maximize transitive dependency

D. To increase dependency

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer B: To reduce redundancy

Normalization reduces repeated data and resulting anomalies.

Why each choice is right or wrong

A - Incorrect: Integrity is protected (PDF 16.)

B - Correct: Redundancy reduction is explicit (PDF 16.)

C - Incorrect: 3NF removes transitive dependence (PDF 16.)

D - Incorrect: Increasing dependency is not the goal (PDF 16.)

Primary note proof

Study-note section FDB19

> Suppose Enrollment repeats StudentName for every course. An update anomaly requires changing every copy of the student's name; missed copies disagree. An insertion anomaly prevents recording a new student until some enrollment exists. A deletion anomaly loses the only record of a student's details when their last enrollment is deleted. Revoke anomaly is not a normalization anomaly.

Study-note section FDB19

> Reduced redundancy can improve integrity and storage use, but extra keys and joins mean neither storage savings nor faster retrieval is guaranteed.

Study-note section FDB19

> Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design.

Related reading (context, not primary note proof)

Study-note section FDB21

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 26-27

> database design technique that minimizes redundancy from a relation or a set of relations. Normalization rules divides larger tables into smaller tables and links them using relationship.

Question sources

Computer Science Exit Exam · 2018 · Question 70

Question 50 | Q00964

For a relation with attributes ID, CITY and NAME, ID is guaranteed unique and non-NULL; CITY and NAME are not unique. Which listed attribute sets are superkeys? Select all that apply.

A. NAME

B. ID

C. CITY

D. CITY, ID

Hint

Test uniqueness and minimality separately when classifying keys.

Answer B, D: B. ID; D. CITY, ID

A superkey is any attribute set guaranteed to identify a tuple; it need not be minimal. ID has the stated uniqueness guarantee, and adding CITY cannot destroy it. Thus ID and (CITY, ID) are superkeys. CITY and NAME individually lack uniqueness guarantees. Only ID is a minimal candidate key among the two qualifying choices, but minimality is not required here.

Why each choice is right or wrong

A - Incorrect: Repeated names are permitted, so NAME is not a guaranteed identifier.

B - Correct: ID is guaranteed unique and identifies each tuple.

C - Incorrect: Repeated cities are permitted, so CITY alone is not a superkey.

D - Correct: The set includes unique ID and therefore retains uniqueness.

Primary note proof

Study-note section FDB07

A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

If ID is unique, both {ID} and {ID,City} are superkeys, but only {ID} is minimal.

Study-note section FDB07

Prefer stable, compact identifiers when practical, but Email, SSN or EmployeeID is a candidate only if business rules guarantee uniqueness.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 7-8

Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key.

Question sources

Computer Science Exit Exam · 2018 · Question 77

Question 51 | Q00982

is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

A. Super key

B. Candidate key

C. Social security key

D. Primary key

Hint

Test uniqueness and minimality separately when classifying keys.

Answer D: Primary key

A primary key is the candidate selected by the designer.

Why each choice is right or wrong

A - Incorrect: Superkey need not be minimal (PDF 11.)

B - Incorrect: Candidate need not be selected (PDF 11.)

C - Incorrect: Example identifier is not this role (PDF 11.)

D - Correct: Exact selected-key definition (PDF 11.)

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 10-11

> Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Question sources

Computer Science Exit Exam · 2018 · Question 95

Question 52 | Q01151

12.
is a collection of concepts thatcan be used to describe the structure of a database.

A. a | Data Architecture

B. i? | Data Model

C. oO Data client

D. | All of the above

Hint

Compare the data models by how they organize records and relationships.

Answer B: i? | Data Model

The defined collection of concepts describing database structure is a data model.

Why each choice is right or wrong

A - Incorrect: Architecture is not the defined term (PDF 7.)

B - Correct: Exact definition (PDF 7.)

C - Incorrect: Client is an application role (PDF 7.)

D - Incorrect: Other choices do not match (PDF 7.)

Primary note proof

Study-note section FDB02

> A data model supplies concepts for structure, relationships, constraints and operations. Conceptual ER models describe entities and relationships. Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents. Object models represent objects with identity, state and behavior. A physical model specifies record layouts and access paths. These are different representation choices, not stages of one query.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 4-5

> A data model a collection of concepts that can be used to describe the structure of a database and provides the necessary means to achieve data abstraction. By structure of a database we mean the

Question sources

Computer Science Exit Exam practice questions · Question 12

Question 53 | Q01152

The description of a database is called

A. a | Database snapshot

B. i? | Database instances

C. @) Database schema

D. | All of the above

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer C: @) Database schema

Schema describes the designed database; instance and snapshot describe current data.

Why each choice is right or wrong

A - Incorrect: Current contents (PDF 7.)

B - Incorrect: Current occurrences (PDF 7.)

C - Correct: Designed structure (PDF 7.)

D - Incorrect: Other choices are different concepts (PDF 7.)

Primary note proof

Study-note section FDB03

> A schema describes names, structures and constraints; an instance, state or snapshot is the data at a particular time. Inserting a student changes the state, while adding a column changes the schema. Metadata is data describing data. The catalog or data dictionary records table definitions, types, constraints and often access privileges and optimizer statistics. A schema diagram draws tables and key references; labels such as database diagram can overlap.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 21-22

> The description of a database is called the database schema, which is specified during database design and is not expected to change frequently. The data in the database at a particular moment

Question sources

Computer Science Exit Exam practice questions · Question 13

Question 54 | Q01153

Which listed terms denote database contents at one particular moment, rather than the database definition? Select all that apply.

A. Database instance

B. Database state / snapshot

C. Database schema

D. None of the above

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer A, B: A. Database instance; B. Database state / snapshot

An instance, state or snapshot describes the actual data at a selected time. Those contents can change after an insertion or update without changing the database definition. A schema describes names, structures and constraints instead. Therefore both the instance option and the state/snapshot option fit, while none of the above cannot be selected because these matching terms are present.

Why each choice is right or wrong

A - Correct: A database instance denotes its data at a particular time.

B - Correct: State and snapshot are synonyms for the same time-specific database contents.

C - Incorrect: A schema describes structure and constraints rather than current data values.

D - Incorrect: Both instance and state/snapshot are provided and correctly fit the definition.

Primary note proof

Study-note section FDB03

> A schema describes names, structures and constraints; an instance, state or snapshot is the data at a particular time. Inserting a student changes the state, while adding a column changes the schema. Metadata is data describing data. The catalog or data dictionary records table definitions, types, constraints and often access privileges and optimizer statistics. A schema diagram draws tables and key references; labels such as database diagram can overlap.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 22-23

> design and is not expected to change frequently. The data in the database at a particular moment of time is called a database state or snapshot. It is also called the current set of occurrences or

Question sources

Computer Science Exit Exam practice questions · Question 14

Question 55 | Q01154

What is the capacity to change the conceptual schema while preserving affected external schemas and application interfaces?

A. Logical data independence

B. Physical data independence

C. Conceptual database

D. All of the above

Hint

Trace which schema levels a proposed change affects.

Answer A: Logical data independence

Logical data independence allows changes to the conceptual schema while preserving the external views and application interfaces that depend on it. Physical data independence instead concerns changing internal storage without changing the conceptual schema. A conceptual database or schema describes the logical structure itself; naming that structure does not express the ability to preserve external interfaces during a change. Consequently A names the requested property and the all-of-the-above choice is false.

Why each choice is right or wrong

A - Correct: Correct: logical data independence preserves external interfaces across conceptual-schema changes.

B - Incorrect: Incorrect: physical data independence concerns internal-storage changes while preserving the conceptual schema.

C - Incorrect: Incorrect: conceptual database names the logical structure, not the property preserving external interfaces when that structure changes.

D - Incorrect: Incorrect: physical independence and the conceptual database are not the requested change-preservation property.

Primary note proof

Study-note section FDB04

> The external level gives each user group a selected view. The conceptual level describes the whole logical database: entity types, relationships, data types and constraints.

Study-note section FDB04

> Physical data independence permits changing internal storage without changing the conceptual schema, for example adding an index. Logical data independence permits conceptual changes while preserving affected external views and application interfaces, for example splitting a table while maintaining a compatible view.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 8 | lines 12-13

1. Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs.

Question sources

Computer Science Exit Exam practice questions · Question 15

Question 56 | Q01155

is language used to define the database conceptual schema

A. a | Data Manipulation Language (DML) x

B. ? | Data Definition Langugae (DDL) v

C. /@| Transaction Query Language (TQL) x

D. | None of the above x

Hint

Classify each SQL command by the kind of change it performs.

Answer B: ? | Data Definition Langugae (DDL) v

Defining a schema is definition rather than manipulation. DDL commands define structure, while DML commands manipulate data.

Why each choice is right or wrong

A - Incorrect: DML covers retrieval and updates, not schema definition. (PDF pp. 26.)

B - Correct: DDL is the definition-language category. (PDF pp. 26.)

C - Incorrect: TQL is not the module's schema language. (PDF pp. 26.)

D - Incorrect: The definition-language choice is present. (PDF pp. 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Computer Science Exit Exam practice questions · Question 16

Question 57 | Q01157

Common functions applied to collections of numeric values: SUM, AVERAGE, COUNT. This function belongs to

A. Ww Grouping x

B. i? | Aggregate functions 4

C. |@| All of the above x

D. | None of the above ><

Hint

Apply grouping before filtering groups and computing aggregate output.

Answer B: Aggregate functions.

SUM, average and COUNT summarize collections into values, which is aggregation. GROUP BY instead partitions rows into groups for those summaries. SQL normally spells the average function AVG; the conceptual use of AVERAGE here still describes aggregation.

Why each choice is right or wrong

A - Incorrect: Grouping partitions rows; it is not the function category containing SUM and COUNT.

B - Correct: These functions summarize a collection of values into a count, total or average.

C - Incorrect: Grouping and aggregate functions are distinct concepts, so all listed categories cannot be correct.

D - Incorrect: Aggregate functions is present and correctly describes the functions in the stem.

Primary note proof

Study-note section FDB32

> COUNT(*) counts rows, COUNT(x) counts non-NULL x values; SUM and AVG summarize non-NULL numeric values. SQL's usual average function is AVG, not AVERAGE. GROUP BY creates one group per distinct grouping value; HAVING tests aggregate groups after WHERE has filtered individual rows.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Computer Science Exit Exam practice questions · Question 18

Question 58 | Q01158

Entity types that do not have key attributes of their own are called

A. a | Entity

B. i? | Strong Entity

C. @) Weak Entity

D. | Foreign Entity

Hint

Check how a dependent entity obtains its complete identity.

Answer C: @) Weak Entity

Weak entities lack their own key attributes, while strong entities have their own keys. Their partial keys identify them only within an owner.

Why each choice is right or wrong

A - Incorrect: Entity is the broader category and does not identify the missing-key subtype. (PDF pp. 10.)

B - Incorrect: Strong entities have their own key, the opposite condition. (PDF pp. 10.)

C - Correct: Weak entity matches the exact definition. (PDF pp. 10.)

D - Incorrect: Foreign entity is not the defined subtype; foreign keys concern table references. (PDF pp. 10.)

Primary note proof

Study-note section FDB08

> A weak entity lacks a complete independent key. Its owner is normally a strong entity, so owner entity set and strong entity set may both describe the associated set. Its partial key, also called discriminator, distinguishes weak entities of the same owner. Identification combines the owner key and partial key.

Study-note section FDB08

> A weak entity is not merely any table with a foreign key.

Study-note section FDB08

> Then (EmployeeID,DependentName) is the key, and EmployeeID is a foreign key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 6-10

> Each entity has attributes, the particular properties that describe it. Entities that do not have key attributes of their own are called weak entity. The regular entity types that have a key attribute are called strong entity types. Weak entities are identified by being related to specific strong entities in combination with one of their attribute values. A weak entity type has a partial key, which is the attribute that can uniquely identify weak entities that are related to the same owner entity.

Question sources

Computer Science Exit Exam practice questions · Question 19

Question 59 | Q01159

The partial key is sometimes called

A. a | Dominant Entity

B. i? | Child Entity Type

C. @) Subordinate Entity Type

D. | Discriminator

Hint

Check how a dependent entity obtains its complete identity.

Answer D: Discriminator.

A partial key distinguishes weak entities belonging to the same owner; the owner key is still needed for global identification. FDB08 explicitly calls that partial key a discriminator. The other choices describe entity roles or types rather than the identifying attribute.

Why each choice is right or wrong

A - Incorrect: An entity role is not the name of the identifying partial-key attribute.

B - Incorrect: A child entity type describes a type of entity rather than its partial identifier.

C - Incorrect: A subordinate entity type is not a synonym for a partial key.

D - Correct: Discriminator is the partial key that distinguishes weak entities within the same owner.

Primary note proof

Study-note section FDB08

> A weak entity lacks a complete independent key. Its owner is normally a strong entity, so owner entity set and strong entity set may both describe the associated set. Its partial key, also called discriminator, distinguishes weak entities of the same owner. Identification combines the owner key and partial key.

Study-note section FDB08

The weak entity participates totally in its identifying relationship.

Study-note section FDB08

Chen notation uses a double rectangle, double identifying diamond and dashed underline for a partial key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 6-13

Each entity has attributes, the particular properties that describe it. Entities that do not have key attributes of their own are called weak entity. The regular entity types that have a key attribute are called strong entity types. Weak entities are identified by being related to specific strong entities in combination with one of their attribute values. A weak entity type has a partial key, which is the attribute that can uniquely identify weak entities that are related to the same owner entity. In ER diagrams, both a weak entity type and its identifying relationship are distinguished by surrounding their boxes and diamonds with double lines. The partial key attribute is underlined with a dashed or dotted line.

Question sources

Computer Science Exit Exam practice questions · Question 20

Question 60 | Q01162

A real-world physical or logical object is called:

A. Attributes

B. Relationship

C. Entity

D. All of the above

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer C: Entity

An entity may have physical existence or conceptual existence. Attributes describe an entity; relationships associate entity types. The stem asks for the object itself.

Why each choice is right or wrong

A - Incorrect: An attribute is a property of the object. (PDF pp. 10, 11.)

B - Incorrect: A relationship associates objects rather than naming the object. (PDF pp. 10, 11.)

C - Correct: Entity covers both physical and conceptual objects. (PDF pp. 10, 11.)

D - Incorrect: The other categories have different roles, so all is false. (PDF pp. 10, 11.)

Primary note proof

Study-note section FDB06

An entity is a distinguishable real or conceptual thing, such as one student or one course. An entity type describes common properties; its current collection of entities is an entity set. A relationship associates entities. In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes. Always inspect the diagram legend.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 3-8

An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course). In ER diagrams, an entity is placed in a rectangle. Each entity has attributes, the particular properties that describe it. Entities that do not have key attributes of their own are called weak entity. The regular entity types that have a key attribute are called strong entity types. Weak entities are identified by being related to specific strong entities

Fundamental of Database Systems module | PDF page 11 | lines 13-15

A relationship type represents the association between entity types. For example, _Enrolled in‘ is a relationship type that exists between entity Student and Course. In ER diagram, relationship is represented by a diamond and connecting the entities with lines.

Question sources

Computer Science Exit Exam practice questions · Question 23

Question 61 | Q01163

In the usual normalization sequence, which is the first normal form that excludes partial dependencies of nonprime attributes on any candidate key?

A. First Normal Form (1NF)

B. Third Normal Form (3NF)

C. Second Normal Form (2NF)

D. Boyce-Codd Normal Form (BCNF)

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer C: Second Normal Form (2NF)

Second normal form is the first stage requiring atomic relations to have no nonprime attribute partially dependent on any candidate key. First normal form addresses atomic values, so it does not impose that dependency rule. Third normal form and BCNF are stronger stages that inherit the absence of partial dependencies, but they are not the first stage introducing it. BCNF specifically requires every nontrivial functional-dependency determinant to be a superkey.

Why each choice is right or wrong

A - Incorrect: Incorrect: 1NF requires atomic values but does not exclude partial dependencies.

B - Incorrect: Incorrect: 3NF inherits 2NF and imposes its additional determinant/prime-attribute condition; it is not the first stage.

C - Correct: Correct: 2NF is the first stage excluding nonprime partial dependencies on candidate keys.

D - Incorrect: Incorrect: BCNF requires every nontrivial determinant to be a superkey; it is stronger than the first stage imposing the partial-dependency rule.

Primary note proof

Study-note section FDB20

> 1NF requires atomic values in the chosen relational domains and no repeating groups in cells.

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Study-note section FDB21

> For every nontrivial FD X→A, 3NF permits it when X is a superkey or A is prime. BCNF requires X to be a superkey. The mnemonic whole key and nothing but the key describes the common single-key case; it cannot replace the full tests when keys overlap. Remove dependencies of nonprime facts on nonkey determinants.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 2-15

> First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF). Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent

attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Computer Science Exit Exam practice questions · Question 24

Question 62 | Q01164

In the usual normalization sequence, which is the first normal form that excludes transitive dependence of nonprime attributes on a key through a nonkey determinant?

A. Third Normal Form (3NF)

B. Fourth Normal Form (4NF)

C. First Normal Form (1NF)

D. All of the above

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer A: Third Normal Form (3NF)

In the stated single-key teaching case, 2NF excludes partial dependencies but can still allow a nonprime fact to depend on the key through a nonkey determinant. The next defining stage is 3NF, summarized by the whole-key-and-nothing-but-the-key mnemonic. BCNF strengthens the determinant rule, and 4NF addresses multivalued dependencies. Stronger forms inherit earlier conditions, so the explicit first-stage scope makes 3NF the unique selection rather than treating inherited properties as rival answers.

Why each choice is right or wrong

A - Correct: Correct: 3NF is the first stage imposing the specified nonprime transitive-dependency restriction in this teaching case.

B - Incorrect: Incorrect: 4NF addresses multivalued dependencies and is a later stage; it is not the first stage imposing the 3NF condition.

C - Incorrect: Incorrect: 1NF concerns atomic values and does not eliminate the specified transitive dependencies.

D - Incorrect: Incorrect: Not all options identify the requested first stage: 1NF does not eliminate the specified transitive dependency and 4NF is later.

Primary note proof

Study-note section FDB20

1NF requires atomic values in the chosen relational domains and no repeating groups in cells.

Study-note section FDB20

2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Study-note section FDB21

For every nontrivial FD X→A, 3NF permits it when X is a superkey or A is prime. BCNF requires X to be a superkey. The mnemonic whole key and nothing but the key describes the common single-key case; it cannot replace the full tests when keys overlap. Remove dependencies of nonprime facts on nonkey determinants.

Study-note section FDB21

4NF addresses nontrivial multivalued dependencies with nonsuperkey determinants; 5NF addresses join dependencies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 2-9

First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF).

Fundamental of Database Systems module | PDF page 18 | lines 5-8

Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Computer Science Exit Exam practice questions · Question 25

Question 63 | Q01217

A conceptual level refers to:

A. Describes the represents global view of the entire database

B. Describes the part of the database that a particular user group views of the data environment

C. Describes how data is actually stored on disk

D. Describes the physical storage structure of the database

Hint

Trace which schema levels a proposed change affects.

Answer A: Describes the represents global view of the entire database

The conceptual schema describes the whole database while hiding storage details. External views show a particular group's part, and internal schema describes storage.

Why each choice is right or wrong

A - Correct: Whole-database structure is conceptual. (PDF pp. 7, 8.)

B - Incorrect: A user-group subset is external. (PDF pp. 7, 8.)

C - Incorrect: Actual storage is internal. (PDF pp. 7, 8.)

D - Incorrect: Physical storage structure is internal. (PDF pp. 7, 8.)

Primary note proof

Study-note section FDB04

> The external level gives each user group a selected view. The conceptual level describes the whole logical database: entity types, relationships, data types and constraints. The internal/physical level describes files, record placement and access paths. These are abstraction levels, distinct from client/server deployment tiers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 25-30

> 2.3 Three-schema Architecture and Data Independence The goal of the three-schema architecture is to separate the user applications from the physical database. In this architecture, schemas can be defined at the following three levels: 1. The internal level has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.

Fundamental of Database Systems module | PDF page 8 | lines 2-7

> 2. The conceptual level has a conceptual schema, which describes the structure of the whole database for users. The conceptual schema hides the details of physical storage structures and

concentrates on describing entities, data types, relationships, user operations, and constraints. 3. The external or view level includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

Question sources

Computer Science model exam 2 · Question 38

Question 64 | Q01226

Suppose you want to design a database for a certain company after you know the need of all users of the database. What is the next step you are expected to do?

A. Logical design

B. Normalization

C. Query Processing

D. Conceptual design

Hint

Identify the design phase from its inputs and intended deliverable.

Answer D: Conceptual design

After collecting users' information requirements, conceptual design develops the ER/EER model that represents those needs. Logical design then maps that model to relations, keys and constraints and checks normalization. Normalization refines dependency structure rather than replacing the initial conceptual modeling step. Query processing evaluates database requests instead of defining the initial enterprise model. D is therefore the next step under the stated sequence, rather than a claim that design can never iterate.

Why each choice is right or wrong

A - Incorrect: Incorrect: logical design maps an already developed conceptual model into relations and constraints.

B - Incorrect: Incorrect: normalization refines relation dependencies during logical design; the initial conceptual model must first be developed.

C - Incorrect: Incorrect: query processing evaluates requests rather than creating the initial model from requirements.

D - Correct: Correct: conceptual design follows the collection of requirements in the stated sequence.

Primary note proof

Study-note section FDB05

First collect business facts, required transactions, reports, volumes, privacy rules and quality targets. Conceptual design turns them into an ER/EER model. Logical design maps the conceptual model to relations, keys and constraints and checks normalization. Physical design chooses file organization, indexes and storage. Implementation creates and loads the actual objects; testing checks requirements.

Related reading (context, not primary note proof)

Study-note section FDB19

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 10-26

Requirement Analysis The most important step in implementing a database system is to find out what is needed i.e what type of a database is required for the business organization, daily volume of data, how much data needs to be stored in the master files etc. In order to collect all this information, a database analyst spends a lot of time within the business organization talking to people, end users and getting acquainted with the day-to-day process. Conceptual Database Design In this phase the database designers will make a decision on the database model that perfectly suits the organization‘s requirement. The database designers will study the documents prepared by the analysis in the requirement analysis stage and then start development of a system model that fulfils the needs. Logical database design (Data model mapping) The conceptual design is translated into internal model which includes mapping of all objects i.e design of tables, indexes, views, transaction, access privileges etc. Physical Design Physical design or Database implementation needs the formation of special storage related constructs. These constructs consist of storage groups, table spaces, data files, tables etc.

Question sources

Computer Science model exam 2 · Question 47

Question 65 | Q01278

All are true about the Database except:

A. Program data independence.

B. Self-describing nature

C. Sharing of data and multi-user transaction processing

D. Support of single views of the data

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer D: Support of single views of the data

The listed characteristics include insulation between data and programs, self-description, sharing, and multiple views. The exception is single views, interpreted as support restricted to one view; a DBMS can of course also display a single view.

Why each choice is right or wrong

A - Incorrect: Program/data insulation is expressly a characteristic. (PDF pp. 5, 6.)

B - Incorrect: Self-description is expressly listed. (PDF pp. 5, 6.)

C - Incorrect: Sharing and multiuser transactions are expressly listed. (PDF pp. 5, 6.)

D - Correct: The stated characteristic is multiple views, rather than restriction to a single view. (PDF pp. 5, 6.)

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB01

> A DBMS centralizes metadata and supports multiple views, authorization, transactions, backup and recovery.

Study-note section FDB04

> The external level gives each user group a selected view.

Related reading (context, not primary note proof)

Study-note section FDB03

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 22-29

> 1.3 Characteristics of the Database Approach The main characteristics of the database approach are Self-describing nature, Data abstraction and Insulation between programs and data, Support of multiple views of the data, sharing of data and multiuser transaction processing.
> Self-describing nature - A fundamental characteristic of the database approach is that the database system contains not only the database itself (data) but also a complete definition or description of the database structure and constraints. This information is called meta-data, and it describes the structure of the database.

Fundamental of Database Systems module | PDF page 6 | lines 2-9

> Data abstraction and Insulation between programs and data- Data Abstraction is a process of hiding unwanted or irrelevant details from the end user. It provides a different view and helps

in achieving data independence which is used to enhance the security of data. Support of multiple views of the data - A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored. Sharing of data and multiuser transaction processing - A multiuser DBMS must allow multiple users to access the database at the same time. This is essential if data for multiple applications is to be integrated and maintained in a single database.

Question sources

Computer Science model exam 2 · Question 99

Question 66 | Q01326

Which one of the following provides the ability to query information from the database and to insert tuples into, delete tuples from, and modify tuples in the database?

A. DML(Data Manipulation Langauge)

B. DDL(Data Definition Langauge)

C. Query

D. Relational Schema

Hint

Classify each SQL command by the kind of change it performs.

Answer A: DML(Data Manipulation Langauge)

DML includes SELECT, INSERT, DELETE, UPDATE and therefore covers both retrieval and tuple changes. The second list is misprinted as Basic DDL commands but sits under the DML heading; the heading and separation resolve that typo.

Why each choice is right or wrong

A - Correct: DML is the category covering all named operations. (PDF pp. 26.)

B - Incorrect: DDL describes structural commands. (PDF pp. 26.)

C - Incorrect: A query is a request, not the complete language category for all modifications. (PDF pp. 26.)

D - Incorrect: A relational schema describes structure rather than performing operations. (PDF pp. 26.)

Primary note proof

Study-note section FDB26

DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 1

Question 67 | Q01327

Which command is used to modifies an existing database object, such as a table?

A. Create

B. Alter

C. Drop

D. Modify

Hint

Distinguish table definition changes from changes to stored rows.

Answer B: ALTER.

ALTER changes the definition of an existing database object. FDB30 demonstrates adding or removing columns and adding constraints with ALTER TABLE. CREATE establishes an object and DROP removes it. MODIFY can occur within some vendor-specific alteration syntax, but it is not the general top-level command requested here.

Why each choice is right or wrong

A - Incorrect: CREATE defines a new object rather than altering the definition of an existing one.

B - Correct: ALTER is the command used to change an existing database object definition.

C - Incorrect: DROP removes the object instead of modifying its existing definition.

D - Incorrect: MODIFY is not the general standalone command; some dialects use it inside ALTER syntax.

Primary note proof

Study-note section FDB30

> ALTER TABLE Employee ADD COLUMN Email VARCHAR(120); adds a column in PostgreSQL-style syntax. ALTER TABLE Employee DROP COLUMN Email; removes it. ALTER TABLE Employee ADD

CONSTRAINT uq_email UNIQUE (Email); adds a named constraint. Use ALTER rather than DALTER, and do not separate column_name and column_type with a comma; validate the resulting SQL before execution.

Study-note section FDB26

DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications.

Study-note section FDB26

Some texts separate SELECT as DQL.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 2

Question 68 | Q01328

Which command is used to modifies records?

A. Insert

B. Update

C. Delete

D. Modify

Hint

Track which rows each data-changing command affects.

Answer B: Update

The module puts UPDATE in DML (PDF p.26). Its UPDATE(Salary) privilege example explicitly permits the Finance user to change Salary values (PDF pp.54-55), grounding UPDATE as the command for modifying existing record values.

Why each choice is right or wrong

A - Incorrect: INSERT is a separate DML operation, illustrated as adding VALUES rows on PDF p.77.

B - Correct: UPDATE is explicitly tied to changing Salary values on PDF p.55.

C - Incorrect: DELETE is a separate operation illustrated with a row predicate on PDF p.77.

D - Incorrect: MODIFY is not the command used for the demonstrated change-value privilege on PDF pp.54-55.

Primary note proof

Study-note section FDB29

> INSERT INTO Hotel(HotelID,HotelName,SubCity) VALUES (1,'Hilton','Yeka'); pairs each named column with a compatible value. Counts and types must match; keys and other constraints still apply. Without a column list, values must follow the table's declared column order.

Study-note section FDB29

> UPDATE without WHERE targets every row.

Study-note section FDB29

> DELETE FROM r; removes every row but leaves r defined.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 10-11

> 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 54 | lines 16-18

> • <privileges> is a list of SELECT <columns>, INSERT <columns>, DELETE, and UPDATE <columns>, or simply ALL

Fundamental of Database Systems module | PDF page 55 | lines 2-5

> UPDATE(Salary) ON Employee TO Finance The user _Finance‘ can view the entire Employee table, and can change Salary values, but cannot change other values or pass on their privilege

Fundamental of Database Systems module | PDF page 77 | lines 14-24

> Example 1: Commit Transaction BEGIN TRANSACTION; DELETE FROM HumanResources.JobCandidate WHERE JobCandidateID = 13; COMMIT; Example 2: Rollback Transaction CREATE TABLE ValueTable (id int); BEGIN TRANSACTION; INSERT INTO ValueTable VALUES(1); INSERT INTO ValueTable VALUES(2); ROLLBACK;

Question sources

Database Fundamentals practice questions · Question 3

Question 69 | Q01329

In the SQLite command-line shell, which command lists the names and files of attached databases?

A. db

B. use

C. .databases

D. database

Hint

Classify each SQL command by the kind of change it performs.

Answer C: C. .databases

The command belongs to the SQLite shell rather than ordinary SQL statement syntax. Its documented spelling is .databases, including the leading dot and final s, and it lists attached database names and files. The corrected C has that spelling. The alternatives db, use and database do not form this SQLite attached-database listing command.

Why each choice is right or wrong

A - Incorrect: db is not the SQLite shell listing command.

B - Incorrect: use does not list SQLite attached names and files.

C - Correct: The .databases command performs the requested listing.

D - Incorrect: database lacks the required shell-command spelling.

Primary note proof

Study-note section FDB26

DELETE removes rows and keeps the table; DROP TABLE removes the table object; ALTER changes its definition. TRUNCATE empties a table using implementation-specific mechanisms. PostgreSQL allows transactional rollback of TRUNCATE, so no rollback information is retained is not a universal SQL property. SQL Server permits DROP DATABASE Sales, suppliers; that comma-list syntax is vendor-specific. SQLite's .databases lists attached names/files in its command-line shell; .database is not the documented command.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 4

Question 70 | Q01330

What is an RDBMS?

A. Database that stores data elements that are not linked

B. Database that accesses data elements that are not linked

C. Database that stores and allows access to data elements that are linked

D. None of the mentioned

Hint

Compare the data models by how they organize records and relationships.

Answer C: Database that stores and allows access to data elements that are linked

A relational DBMS manages a database represented by relations/tables, with references maintaining relationships between tuples. Among these broad descriptions C is closest, although linked data alone is not a complete definition distinguishing relational from other models.

Why each choice is right or wrong

A - Incorrect: A description excluding links omits the relationships modeled by references. (PDF pp. 5, 13, 14.)

B - Incorrect: Accessing only unlinked elements also omits those relationships. (PDF pp. 5, 13, 14.)

C - Correct: Storage and access to related elements agrees with tables and relational constraints. (PDF pp. 5, 13, 14.)

D - Incorrect: C supplies the compatible broad description, though it is not a formal definition. (PDF pp. 5, 13, 14.)

Primary note proof

Study-note section FDB02

> A data model supplies concepts for structure, relationships, constraints and operations. Conceptual ER models describe entities and relationships. Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents. Object models represent objects with identity, state and behavior. A physical model specifies record layouts and access paths. These are different representation choices, not stages of one query.

Study-note section FDB03

> A relation is a set of tuples, represented as a table.

Study-note section FDB03

Classical relations have distinct tuples, atomic cells and domains for attributes; row order has no meaning.

Related reading (context, not primary note proof)

Study-note section FDB15

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 14-16

Database Management System (DBMS) is then a tool for creating and managing databases. DBMS is a software that facilities the processes of defining, constructing, manipulating, and sharing databases. Example: MySQL server, MS ACESS, Microsoft SQL server.

Fundamental of Database Systems module | PDF page 13 | lines 18-27

3.6 The Relational Database Model The relational model represents the database as a collection of relations. A table is represented with columns and rows. Each table is an entity in the ER diagram. Each row is known as a tuple. Each table of the column has a name or attribute of the entity. 3.7 Relational Model Concepts in DBMS Attribute: Each column in a Table. Attributes are the properties which define a relation Tables - In the Relational model, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes. Tuple - It is a single row of a table, which contains a single record.

Fundamental of Database Systems module | PDF page 14 | lines 21-26

Referential Integrity Constraints - The Referential integrity constraints is specified between two relations or tables and used to maintain the consistency among the tuples in two relations. Referential Integrity constraints in DBMS are based on the concept of Foreign Keys. A foreign key is an important attribute of a relation which should be referred to in other relationships. Referential integrity constraint state happens where relation refers to a key attribute of a different or same relation. However, that key element must exist in the table.

Question sources

Database Fundamentals practice questions · Question 5

Question 71 | Q01331

What is a relation in RDBMS?

A. Key

B. Table

C. Row

D. Data Types

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer B: Table

The relational model represents relations in table format. A tuple is one row; a key identifies tuples rather than being the relation itself.

Why each choice is right or wrong

A - Incorrect: A key is an identifying attribute set. (PDF pp. 13.)

B - Correct: A relation is represented as a table. (PDF pp. 13.)

C - Incorrect: A row is a tuple within the relation. (PDF pp. 13.)

D - Incorrect: Data types constrain values rather than constituting a relation. (PDF pp. 13.)

Primary note proof

Study-note section FDB03

> A relation is a set of tuples, represented as a table. A tuple/row/record contains related attribute values; an attribute/column/field describes a property. Degree counts columns; cardinality counts rows. Classical relations have distinct tuples, atomic cells and domains for attributes; row order has no meaning. Column order has no logical meaning when attributes are identified by name. SQL tables can contain duplicate rows unless constrained.

Study-note section FDB07

> A superkey uniquely identifies a tuple.

Study-note section FDB07

> A primary key is one selected candidate key; remaining candidates are alternate keys.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 13 | lines 18-27

> 3.6 The Relational Database Model The relational model represents the database as a collection of relations. A table is represented with columns and rows. Each table is an entity in the ER diagram. Each row is known as a tuple. Each table of the column has a name or attribute of the entity. 3.7 Relational Model Concepts in DBMS Attribute: Each column in a Table. Attributes are the properties which define a relation Tables - In the Relational model, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes. Tuple - It is a single row of a table, which contains a single record.

Question sources

Database Fundamentals practice questions · Question 6

Question 72 | Q01332

Relational Algebra is a _____ query language that takes two relations as input and produces another relation as an output of the query.

A. Relational

B. Structural

C. Procedural

D. Fundamental

Hint

Distinguish a declarative condition from a prescribed evaluation procedure.

Answer C: Procedural

Relational algebra is procedural because it specifies a sequence of operations. The stem overgeneralizes arity: operations can take one or more input relations, not always two.

Why each choice is right or wrong

A - Incorrect: Relational describes the model but misses the procedural/calculus distinction. (PDF pp. 24, 24.)

B - Incorrect: Structural is not the stated classification. (PDF pp. 24, 24.)

C - Correct: Procedural is expressly justified by the required sequence. (PDF pp. 24, 24.)

D - Incorrect: Fundamental is not the procedural classification used in the definition. (PDF pp. 24, 24.)

Primary note proof

Study-note section FDB36

> Relational algebra specifies a sequence of operations and is procedural at the logical level. Relational calculus specifies a predicate describing desired results and is declarative/nonprocedural. A tuple variable ranges over tuples; a domain variable ranges over individual attribute values. Safe/range-restricted expressions produce results bounded by the database's active domain.

Study-note section FDB35

> Rename ρ changes relation or attribute names.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 5-9

> Relational algebra is the basic set of operations for the relational model. These operations enable a user to specify basic retrieval requests (or queries). The result of an operation is a new relation, which may have been formed from one or more input relations. The algebra operations thus

produce new relations. Since each operation returns a relation, operations can be composed (Algebra is —closed‖.)

Fundamental of Database Systems module | PDF page 24 | lines 21-26

In a calculus expression, there is no order of operations to specify how to retrieve the query result. A calculus expression specifies only what information the result should contain. This is the main distinguishing feature between relational algebra and relational calculus. Relational calculus is considered to be a nonprocedural language. This differs from relational algebra, where we must write a sequence of operations to specify a retrieval request; hence relational algebra can be considered as a procedural way of stating a query.

Question sources

Database Fundamentals practice questions · Question 7

Question 73 | Q01333

Which of the following is a fundamental operation in relational algebra?

A. Set intersection

B. Natural join

C. Assignment

D. None of the mentioned

Hint

Use the operator convention stated in the question and distinguish membership from derivability.

Answer A: Set intersection

Under the stated Basic operations list, intersection is basic. A smaller primitive basis may instead treat intersection as derived; the answer uses the stated list.

Why each choice is right or wrong

A - Correct: Intersection is explicitly under Basic operations. (PDF pp. 24.)

B - Incorrect: Natural join is absent from this basic list. (PDF pp. 24.)

C - Incorrect: Assignment is absent from this basic list. (PDF pp. 24.)

D - Incorrect: False because intersection is listed as basic. (PDF pp. 24.)

Primary note proof

Study-note section FDB-S01

A list of basic relational-algebra operators is a teaching convention, distinct from a claim that every listed operation is logically indispensable. The course module uses selection, projection, Cartesian product, set difference, union and intersection as its basic-operation list. Natural join and assignment are not members of that particular list.

Study-note section FDB-S01

A smaller primitive basis may therefore omit intersection even when it is introduced as a basic operation.

Study-note section FDB-S01

Under set semantics, union includes tuples from either compatible input, intersection retains tuples common to both, and difference keeps first-input tuples absent from the second.

Related reading (context, not primary note proof)

Study-note section FDB35

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Question sources

Database Fundamentals practice questions · Question 8

Question 74 | Q01334

Which of the following is used to denote the selection operation in relational algebra?

A. Pi (Greek)

B. Sigma (Greek)

C. Lambda (Greek)

D. Omega (Greek)

Hint

Apply each algebra operator to the input relations step by step.

Answer B: Sigma (Greek)

Selection filters tuples and is conventionally denoted by sigma, σ. Projection keeps attributes and is denoted by pi, π. Lambda and omega are different Greek symbols and therefore cannot identify the requested sigma operation under this notation. This is a question about the operator's conventional symbol, rather than whether another symbol could be assigned a new meaning in a custom notation. B is the unique selection symbol among the choices.

Why each choice is right or wrong

A - Incorrect: Incorrect: pi, π, denotes projection, which retains attributes.

B - Correct: Correct: sigma, σ, denotes selection, which filters tuples.

C - Incorrect: Incorrect: lambda, λ, is a different symbol from the conventional selection symbol σ.

D - Incorrect: Incorrect: omega, ω, is a different symbol from the conventional selection symbol σ.

Primary note proof

Study-note section FDB35

> Selection σ filters tuples; projection π retains attributes and removes duplicate tuples under set semantics.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

> Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Question sources

Database Fundamentals practice questions · Question 9

Question 75 | Q01335

For union-compatible relations, which listed names denote the minus operation R−S, returning tuples in R but not S? Select all that apply.

A. Union

B. Set-difference

C. Difference

D. Intersection

Hint

Use the operator convention stated in the question and distinguish membership from derivability.

Answer B, C: B. Set-difference; C. Difference

R−S tests membership asymmetrically: a tuple must occur in R and must not occur in S. Difference and set-difference are names for this same operator, so both should be selected. Union combines membership from either input, whereas intersection requires membership in both. Swapping the inputs to difference can change the result because the exclusion direction matters.

Why each choice is right or wrong

A - Incorrect: Union combines compatible input tuples rather than retaining only first-input tuples absent from the second.

B - Correct: Set-difference returns tuples in the first relation that do not occur in the second.

C - Correct: Difference is the same minus operator as set-difference.

D - Incorrect: Intersection keeps tuples common to both relations, the opposite membership condition.

Primary note proof

Study-note section FDB-S01

> Under set semantics, union includes tuples from either compatible input, intersection retains tuples common to both, and difference keeps first-input tuples absent from the second. Intersection can be expressed using difference, while a matching join can be expressed as selection over a Cartesian product. A smaller primitive basis may therefore omit intersection even when it is introduced as a basic operation.

Study-note section FDB35

> Union, difference and intersection require union-compatible schemas.

Study-note section FDB35

> Rename ρ changes relation or attribute names.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

> Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Question sources

Database Fundamentals practice questions · Question 10

Question 76 | Q01336

Which one of the following is used to define the structure of the relation, deleting relations and relating schemas?

A. DML(Data Manipulation Langauge)

B. DDL(Data Definition Langauge)

C. Query

D. Relational Schema

Hint

Classify each SQL command by the kind of change it performs.

Answer B: DDL(Data Definition Langauge)

Defining and deleting relation structures are schema operations in DDL. DML operates on their data. Relating schemas here is interpreted as defining schema structure rather than retrieving joins.

Why each choice is right or wrong

A - Incorrect: DML is for data operations. (PDF pp. 26.)

B - Correct: DDL is the schema-definition category. (PDF pp. 26.)

C - Incorrect: A query request is not the schema language category. (PDF pp. 26.)

D - Incorrect: A schema is a description, not a language for changing it. (PDF pp. 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 11

Question 77 | Q01337

CREATE TABLE employee (name VARCHAR(80), id INTEGER); What kind of SQL statement is this?

A. DML

B. DDL

C. View

D. Integrity constraint

Hint

Classify each SQL command by the kind of change it performs.

Answer B: DDL

CREATE TABLE defines a new table and its columns, so it is a data-definition-language statement. DML instead retrieves or changes stored rows. A view is a query-based logical object, while this statement creates a base table. Integrity constraints can be included inside a table definition, but this whole statement is not merely a constraint expression. VARCHAR has the stated length in this table definition.

Why each choice is right or wrong

A - Incorrect: Incorrect: DML acts on stored data, whereas this statement defines a table.

B - Correct: Correct: creating a table and declaring its columns is DDL.

C - Incorrect: Incorrect: CREATE TABLE defines a table rather than a view.

D - Incorrect: Incorrect: the statement defines an entire table, not just an integrity-constraint expression.

Primary note proof

Study-note section FDB26

DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Study-note section FDB30

CREATE TABLE Employee (EmpID INTEGER PRIMARY KEY, Name VARCHAR(80)); declares two columns. The name and its type stay together; the comma separates whole column declarations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 12

Question 78 | Q01338

SELECT * FROM employee
What type of statement is this?

A. DML

B. DDL

C. View

D. Integrity constraint

Hint

Classify each SQL command by the kind of change it performs.

Answer A: DML

SELECT is in the list under the DML heading. The line's repeated DDL word is a typographical error in that DML subsection, as shown by the preceding separate DDL list.

Why each choice is right or wrong

A - Correct: SELECT is classified under DML. (PDF pp. 26.)

B - Incorrect: SELECT is not in the preceding DDL list. (PDF pp. 26.)

C - Incorrect: No view is defined by this standalone retrieval. (PDF pp. 26.)

D - Incorrect: The retrieval is not an integrity constraint. (PDF pp. 26.)

Primary note proof

Study-note section FDB26

DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Study-note section FDB33

CREATE VIEW CSStudent AS SELECT StudentID,Name FROM Student WHERE Major = 'CS'; stores a query definition and presents a virtual table.

Study-note section FDB33

An ordinary view does not necessarily store its result; a materialized view does.

Related reading (context, not primary note proof)

Study-note section FDB27

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 13

Question 79 | Q01339

The basic data type char(n) is a _____ length character string and varchar(n) is _____ length character.

A. Fixed, equal

B. Equal, variable

C. Fixed, variable

D. Variable, equal

Hint

Relate the value domain and access requirements to storage and type choices.

Answer C: Fixed, variable

The attribute-type section directly defines CHAR as fixed-length and VARCHAR as variable-length. Neither blank means equal length.

Why each choice is right or wrong

A - Incorrect: Fixed is right for CHAR but equal is wrong for VARCHAR. (PDF pp. 19.)

B - Incorrect: Variable is right for VARCHAR but equal is not the stated CHAR class. (PDF pp. 19.)

C - Correct: Fixed followed by variable matches both definitions. (PDF pp. 19.)

D - Incorrect: Both descriptions miss the given definitions. (PDF pp. 19.)

Primary note proof

Study-note section FDB23

> CHAR(n) is fixed length; VARCHAR(n) has variable length up to a limit. DATE represents a date and BLOB binary content such as images. INTEGER represents whole numbers; DECIMAL(p,s) represents exact decimal values, making it suitable for money when scale is specified. FLOAT/REAL/DOUBLE are approximate; names, widths and precision depend on the DBMS. Numeric type widths depend on the DBMS and type; do not assume one universal width.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 19 | lines 11-15

> 5.2 Choosing Attribute Data Types CHAR - fixed-length character VARCHAR() - variable-length characters DATE - actual date BLOB - binary large object (good for graphics images, audio, sound clips, etc.)

Question sources

Database Fundamentals practice questions · Question 14

Question 80 | Q01340

To remove a relation from an SQL database, we use the _____ comman
d.

A. Delete

B. Purge

C. Remove

D. Drop table

Hint

Classify each SQL command by the kind of change it performs.

Answer D: DROP TABLE.

Removing a relation means removing the table object, rather than just its tuples. FDB26 distinguishes DROP TABLE from DELETE: DELETE preserves the table definition while removing rows. DROP TABLE removes the object itself. PURGE and REMOVE are not the general SQL command specified here for this task.

Why each choice is right or wrong

A - Incorrect: DELETE removes tuples while preserving the relation definition, so it does not remove the relation.

B - Incorrect: PURGE is not the general table-removal command established here.

C - Incorrect: REMOVE is not the SQL command used here to remove a relation object.

D - Correct: DROP TABLE removes the table object, including its relation definition.

Primary note proof

Study-note section FDB26

> DELETE removes rows and keeps the table; DROP TABLE removes the table object; ALTER changes its definition. TRUNCATE empties a table using implementation-specific mechanisms. PostgreSQL allows transactional rollback of TRUNCATE, so no rollback information is retained is not a universal SQL property. SQL Server permits DROP DATABASE Sales, suppliers; that comma-list syntax is vendor-specific. SQLite's .databases lists attached names/files in its command-line shell; .database is not the documented command.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

> 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 15

Question 81 | Q01341

Assume constraints permit the deletion. Which descriptions apply to executing DELETE FROM r; on an existing table r? Select all that apply.

A. Remove relation

B. Clear relation entries

C. Delete fields

D. Delete rows

Hint

Track which rows each data-changing command affects.

Answer B, D: B. Clear relation entries; D. Delete rows

Without WHERE, DELETE targets every row in r. If the operation succeeds, the table's entries are cleared but its column definitions and table object remain. Clear relation entries and delete rows describe this same effect. Removing the relation itself requires a schema operation such as DROP TABLE, while deleting fields would change columns rather than row contents.

Why each choice is right or wrong

A - Incorrect: The table definition remains; DELETE does not remove the relation object.

B - Correct: Without WHERE, every relation entry is removed while the table remains available.

C - Incorrect: Fields are columns; DELETE removes rows rather than column definitions.

D - Correct: Every row is targeted because the statement contains no filtering WHERE clause.

Primary note proof

Study-note section FDB29

> INSERT INTO Hotel(HotelID,HotelName,SubCity) VALUES (1,'Hilton','Yeka'); pairs each named column with a compatible value. Counts and types must match; keys and other constraints still apply. Without a column list, values must follow the table's declared column order.

Study-note section FDB29

> Both clear relation entries and delete rows describe that effect.

Study-note section FDB29

> DELETE FROM r; removes every row but leaves r defined.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 10-11

> 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 16

Question 82 | Q01342

INSERT INTO instructor VALUES (10211, 'Smith', 'Biology', 66000);
What type of statement is this?

A. Query

B. DML

C. Relational

D. DDL

Hint

Classify each SQL command by the kind of change it performs.

Answer B: DML

INSERT belongs to the DML list. Classification follows from the operation name; it does not require assuming an instructor schema or proving that these particular values satisfy its constraints.

Why each choice is right or wrong

A - Incorrect: Query is a generic request rather than the displayed language category. (PDF pp. 26.)

B - Correct: INSERT is expressly a DML command. (PDF pp. 26.)

C - Incorrect: Relational names a model, not this statement class. (PDF pp. 26.)

D - Incorrect: INSERT is not in the DDL list. (PDF pp. 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Study-note section FDB29

> Without a column list, values must follow the table's declared column order.

Study-note section FDB29

INSERT INTO Hotel(HotelID,HotelName,SubCity) VALUES (1,'Hilton','Yeka'); pairs each named column with a compatible value.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 17

Question 83 | Q01343

Which joins discard unmatched tuples from both inputs? Interpret Natural join as an ordinary natural inner join. Select all that apply.

A. Left outer join

B. Right outer join

C. Inner join

D. Natural join

Hint

Compute matching pairs, then account for any preserved unmatched rows.

Answer C, D: C. Inner join; D. Natural join

An inner join returns only pairs that satisfy its matching condition, so unmatched tuples from either input disappear. An ordinary natural join is an inner equality join using all common column names and has the same preservation behavior. Left and right outer joins instead preserve unmatched tuples from their designated side by adding NULLs for the other side.

Why each choice is right or wrong

A - Incorrect: A left outer join preserves unmatched left rows, filling right-side columns with NULL.

B - Incorrect: A right outer join preserves unmatched right rows, filling left-side columns with NULL.

C - Correct: An inner join returns matching pairs only, so unmatched tuples do not survive.

D - Correct: An ordinary natural join is an inner equality join and discards unmatched tuples.

Primary note proof

Study-note section FDB31

An inner join returns matching pairs satisfying ON. An equijoin uses equality; a theta join permits another comparison. A natural join is ordinarily an inner equality join on every same-named column and merges those duplicate columns. JOIN ... USING (DeptID) specifies shared join columns explicitly. CROSS JOIN forms every pair without a matching condition.

Study-note section FDB31

LEFT OUTER JOIN also preserves unmatched left rows, padding right columns with NULL; RIGHT preserves right rows and FULL preserves both.

Study-note section FDB31

Therefore inner and natural inner joins both discard unmatched rows.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Fundamental of Database Systems module | PDF page 28 | lines 17-24

Query 2 (Better) (position=„Manager“)^(city=„London“) (Staff Staff.branchNo = Branch.branchNo Branch) - Again requires (1000+50) disk accesses to read from Staff and Branch - Joins Staff and Branch on branchNo with 1000 tuples (1 employee : 1 branch) - Requires (1000) disk access to read in joined relation and check predicate Total Work = (1000+50) + 2*(1000) = 3050 I/O operations

Question sources

Database Fundamentals practice questions · Question 18

Question 84 | Q01344

What type of join is needed when you wish to include rows that do not have matching values?

A. Equi-join

B. Natural join

C. Outer join

D. All of the mentioned

Hint

Compute matching pairs, then account for any preserved unmatched rows.

Answer C: Outer join.

An outer join retains unmatched rows from one or both inputs and fills the missing side with NULL. FDB31 contrasts that behavior with ordinary matching-only joins. Equijoin describes an equality condition, and an ordinary natural join is an inner equality join; neither alone guarantees unmatched-row preservation. Therefore all of the mentioned is false.

Why each choice is right or wrong

A - Incorrect: An equality condition alone does not retain rows that fail to match.

B - Incorrect: An ordinary natural join uses matching shared-column values and discards unmatched rows.

C - Correct: Outer joins preserve unmatched rows from the left, right or both inputs.

D - Incorrect: The other named joins do not all guarantee preservation of unmatched rows.

Primary note proof

Study-note section FDB31

> An inner join returns matching pairs satisfying ON. An equijoin uses equality; a theta join permits another comparison. A natural join is ordinarily an inner equality join on every same-named column and merges those duplicate columns. JOIN ... USING (DeptID) specifies shared join columns explicitly. CROSS JOIN forms every pair without a matching condition.

Study-note section FDB31

> Therefore inner and natural inner joins both discard unmatched rows.

Study-note section FDB31

> LEFT OUTER JOIN also preserves unmatched left rows, padding right columns with NULL; RIGHT preserves right rows and FULL preserves both.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

> Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Fundamental of Database Systems module | PDF page 28 | lines 17-24

> Query 2 (Better) (position=„Manager“)^(city=„London“) (Staff Staff.branchNo = Branch.branchNo Branch) - Again requires (1000+50) disk accesses to read from Staff and Branch - Joins Staff and Branch on branchNo with 1000 tuples (1 employee : 1 branch) - Requires (1000) disk access to

read in joined relation and check predicate Total Work = (1000+50) + 2*(1000) = 3050 I/O operations

Question sources

Database Fundamentals practice questions · Question 19

Question 85 | Q01345

How many input relations does one ordinary binary join operator combine? Count operator inputs, not distinct base tables used by a complete query.

A. One

B. Two

C. Three

D. All of the mentioned

Hint

Compute matching pairs, then account for any preserved unmatched rows.

Answer B: B. Two

One binary join operator combines two input relations. Either input may itself be the output of an earlier join, so the complete query can involve more base tables. A self-join can refer twice to one base table, but still supplies two relation occurrences to the operator. The question fixes the count at the operator level, making two the unique choice.

Why each choice is right or wrong

A - Incorrect: One distinct base table can support a self-join, but the operator still has two inputs.

B - Correct: A binary join operator has exactly two inputs.

C - Incorrect: Three base tables require a chain of binary joins, not one operator.

D - Incorrect: The input count of the specified binary operator is fixed, so all counts do not apply.

Primary note proof

Study-note section FDB31

An inner join returns matching pairs satisfying ON. An equijoin uses equality; a theta join permits another comparison. A natural join is ordinarily an inner equality join on every same-named column and merges those duplicate columns. JOIN ... USING (DeptID) specifies shared join columns explicitly. CROSS JOIN forms every pair without a matching condition.

Study-note section FDB31

Each ordinary join operator combines two inputs, but its inputs can be earlier joins, so one query can join many tables.

Study-note section FDB31

Distinguish base-table count, table occurrences and binary operator arity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 5-9

Relational algebra is the basic set of operations for the relational model. These operations enable a user to specify basic retrieval requests (or queries). The result of an operation is a new relation, which may have been formed from one or more input relations. The algebra operations thus produce new relations. Since each operation returns a relation, operations can be composed (Algebra is —closed‖.)

Fundamental of Database Systems module | PDF page 28 | lines 17-24

Query 2 (Better) (position=„Manager“)^(city=„London“) (Staff Staff.branchNo = Branch.branchNo Branch) - Again requires (1000+50) disk accesses to read from Staff and Branch - Joins Staff and Branch on branchNo with 1000 tuples (1 employee : 1 branch) - Requires (1000) disk access to read in joined relation and check predicate Total Work = (1000+50) + 2*(1000) = 3050 I/O operations

Question sources

Database Fundamentals practice questions · Question 20

Question 86 | Q01346

Which option gives the complete set of valid SQL join forms among the listed alternatives? Choose the most complete single answer.

A. Cross join

B. Natural join

C. Join with USING clause

D. All of the mentioned

Hint

Compute matching pairs, then account for any preserved unmatched rows.

Answer D: All of the mentioned

CROSS JOIN forms every pair of input rows without a matching predicate. NATURAL JOIN matches all same-named columns and merges the duplicate join columns. JOIN USING explicitly names shared equality columns. All three are valid join forms, although they do not all express a join condition in the same way. The revised stem requests the complete single answer, so the aggregate D is selected; each individually true alternative is incomplete on its own.

Why each choice is right or wrong

A - Incorrect: Incorrect as the complete selection: CROSS JOIN is a valid join form without a matching condition, but B and C also qualify.

B - Incorrect: Incorrect as the complete selection: NATURAL JOIN is valid and matches same-named columns, but A and C also qualify.

C - Incorrect: Incorrect as the complete selection: JOIN USING is valid and names shared equality columns, but A and B also qualify.

D - Correct: Correct: all three named constructs are valid SQL join forms, so the aggregate is the complete answer.

Primary note proof

Study-note section FDB31

> An inner join returns matching pairs satisfying ON. An equijoin uses equality; a theta join permits another comparison. A natural join is ordinarily an inner equality join on every same-named column and merges those duplicate columns. JOIN ... USING (DeptID) specifies shared join columns explicitly. CROSS JOIN forms every pair without a matching condition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

> Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Fundamental of Database Systems module | PDF page 28 | lines 17-24

> Query 2 (Better) (position=„Manager“)^(city=„London“) (Staff Staff.branchNo = Branch.branchNo Branch) - Again requires (1000+50) disk accesses to read from Staff and Branch - Joins Staff and Branch on branchNo with 1000 tuples (1 employee : 1 branch) - Requires (1000) disk access to read in joined relation and check predicate Total Work = (1000+50) + 2*(1000) = 3050 I/O operations

Question sources

Database Fundamentals practice questions · Question 21

Question 87 | Q01347

The operation which is not considered a basic operation of relational algebra is

A. Join

B. Selection

C. Union

D. Cross product

Hint

Use the operator convention stated in the question and distinguish membership from derivability.

Answer A: Join

The stated basic-operation list contains selection, union, cross-product, projection, difference, and intersection. Join is not included, even though it appears in later execution-plan examples.

Why each choice is right or wrong

A - Correct: Join is absent from the basic list and is the requested exception. (PDF pp. 24.)

B - Incorrect: Selection is explicitly basic. (PDF pp. 24.)

C - Incorrect: Union is explicitly basic. (PDF pp. 24.)

D - Incorrect: Cross-product is explicitly basic. (PDF pp. 24.)

Primary note proof

Study-note section FDB-S01

> A list of basic relational-algebra operators is a teaching convention, distinct from a claim that every listed operation is logically indispensable. The course module uses selection, projection, Cartesian product, set difference, union and intersection as its basic-operation list. Natural join and assignment are not members of that particular list.

Study-note section FDB-S01

> A smaller primitive basis may therefore omit intersection even when it is introduced as a basic operation.

Study-note section FDB-S01

> Intersection can be expressed using difference, while a matching join can be expressed as selection over a Cartesian product.

Related reading (context, not primary note proof)

Study-note section FDB35

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

> Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Question sources

Database Fundamentals practice questions · Question 22

Question 88 | Q01348

A many to many relationship between two entities usually results in how many tables?

A. One

B. Two

C. Three

D. Four

Hint

Identify relationship degree and cardinality before choosing a relational mapping.

Answer C: Three tables: two entity tables and one association table.

Each of the two entity types normally has its own relation. Their many-to-many relationship requires an additional association relation holding references to both entities and any relationship attributes. FDB12 illustrates Student, Course and Enrollment. Thus the usual total is three, rather than only the one new association table.

Why each choice is right or wrong

A - Incorrect: One counts only the association table and omits the two participating entity tables.

B - Incorrect: Two entity tables alone do not record arbitrary many-to-many pairs.

C - Correct: The two entity tables plus their association table produce the usual total of three.

D - Incorrect: No fourth table is required by the ordinary binary many-to-many mapping.

Primary note proof

Study-note section FDB12

> A binary M:N relationship needs an association table in addition to the two entity tables. Enrollment(StudentID,CourseID,Grade) references Student and Course; (StudentID,CourseID) is a key if one enrollment per pair is allowed. Repeated attempts require Term or AttemptNo in the key. This ordinarily produces three tables.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 12 | lines 2-3

> Many to many (M-M) - When entities in all entity sets can take part more than once in the relationship cardinality is many to many.

Fundamental of Database Systems module | PDF page 12 | lines 8-24

> 3.4 Mapping ER-models to relational tables 1. For each strong entity type E, create a relation R that includes all the attributes of E. 2. Choose one of the key attributes of E as the primary key for R. For each weak entity type W with owner entity type E (which is a strong entity), create a relation R & include all attributes of ‗W‘ as attributes of R. Also, include the primary key attribute

of the relation that correspond to the owner entity type as foreign key attributes of R . 3. For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. Foreign Key approach: Choose one of the relations-say S-and include a foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S. 4. For each binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type. Include the primary key of the relation T as foreign key in S that represents the other entity type participating in R. Include any attributes of the 1:N relation type as attributes of S. 5. For each n-ary relationship type R, where n>2, create a new relationship S to represent R. Include the primary keys of the relations that represent the participating entity types as foreign key attributes in S. Also include any attributes of the n-ary relationship type as attributes of S.

Question sources

Database Fundamentals practice questions · Question 23

Question 89 | Q01349

An _____ is a set of entities of the same type that share the same properties, or attributes.

A. Entity set

B. Attribute set

C. Relation set

D. Entity model

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer A: Entity set.

FDB06 distinguishes an individual entity from the type describing its properties and the set containing the current entities of that type. The stem asks for that collection of similarly described entities, which is an entity set. Attributes describe properties; a model describes representation rather than the current collection itself.

Why each choice is right or wrong

A - Correct: An entity set is the collection of entities belonging to one common entity type.

B - Incorrect: An attribute set groups properties, rather than the entities possessing those properties.

C - Incorrect: Relation set is not the ER term for the stated collection of entities.

D - Incorrect: An entity model describes structures, rather than naming the current set of entity instances.

Primary note proof

Study-note section FDB06

An entity is a distinguishable real or conceptual thing, such as one student or one course. An entity type describes common properties; its current collection of entities is an entity set. A relationship associates entities. In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes. Always inspect the diagram legend.

Study-note section FDB03

A relation is a set of tuples, represented as a table.

Study-note section FDB03

A tuple/row/record contains related attribute values; an attribute/column/field describes a property.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 2-9

An important constraint on the entities of an entity type is the key or uniqueness constraint on attributes. An entity type usually has one or more attributes whose values are distinct for each individual entity in the entity set. Such an attribute is called a key attribute, and its values can be used to identify each entity uniquely. Key Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key.

Fundamental of Database Systems module | PDF page 11 | lines 16-18

Degree of a relationship The number of different entity sets participating in a relationship set is called as degree of a relationship set.

Question sources

Database Fundamentals practice questions · Question 24

Question 90 | Q01350

Entity is a _____

A. Object of relation

B. Present working model

C. Thing in real world

D. Model of relation

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer C: Thing in real world

An entity is an object with physical or conceptual existence. It need not be a physical thing; the real-world thing option includes conceptual objects such as a course.

Why each choice is right or wrong

A - Incorrect: Object of relation is not the module's definition. (PDF pp. 10.)

B - Incorrect: A present working model is not an entity definition. (PDF pp. 10.)

C - Correct: Thing in the real world matches physical or conceptual objects. (PDF pp. 10.)

D - Incorrect: A model describes entities rather than being the object described. (PDF pp. 10.)

Primary note proof

Study-note section FDB06

> An entity is a distinguishable real or conceptual thing, such as one student or one course. An entity type describes common properties; its current collection of entities is an entity set. A relationship associates entities. In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes. Always inspect the diagram legend.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 3-8

> An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course). In ER diagrams, an entity is placed in a rectangle. Each entity has attributes, the particular properties that describe it. Entities that do not have key attributes of their own are called weak entity. The regular entity types that have a key attribute are called strong entity types. Weak entities are identified by being related to specific strong entities

Question sources

Database Fundamentals practice questions · Question 25

Question 91 | Q01351

E-R modeling technique is a :

A. Top-down approach

B. Bottom-up approach

C. Left-right approach

D. None of the above

Hint

Identify the design phase from its inputs and intended deliverable.

Answer A: Top-down is the usual conceptual ER approach.

FDB05 describes conceptual design as usually starting with an enterprise view and working downward into entity and relationship details. This is the usual top-down classification of ER modeling. Bottom-up refinement is associated here with normalization from dependencies. This is a useful conventional approach, rather than a claim that every ER modeling activity must follow one direction.

Why each choice is right or wrong

A - Correct: Conceptual ER modeling usually moves from an overall enterprise view toward detailed entities and relationships.

B - Incorrect: Bottom-up dependency refinement describes normalization here, rather than the usual ER conceptual approach.

C - Incorrect: Left-right is not a database design methodology identified here.

D - Incorrect: Top-down is an available and appropriate description of the usual conceptual ER approach.

Primary note proof

Study-note section FDB05

> First collect business facts, required transactions, reports, volumes, privacy rules and quality targets. Conceptual design turns them into an ER/EER model. Logical design maps the conceptual model to relations, keys and constraints and checks normalization. Physical design chooses file organization, indexes and storage. Implementation creates and loads the actual objects; testing checks requirements.

Study-note section FDB05

> Conceptual design usually proceeds from a whole enterprise view downward; normalization can refine structures from dependencies upward.

Study-note section FDB05

> These are useful approaches rather than mandatory directions for every modeling activity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 3-6

> 3.1 Database Modeling High-level conceptual data models provide concepts for presenting data in ways that are close to the way people perceive data. A typical example is the entity relationship model, which uses main concepts like entities, attributes and relationships.

Fundamental of Database Systems module | PDF page 9 | lines 27-31

> 3.3 ERD (Entity Relationship Diagram) ERD a graphical technique for representing a database schema. It is a modeling tool used to depict graphically a DB design before it is actually implemented. It shows the various entities being modeled and the important relationships among them. ERD has three basic components Entity, Relationship and Attribute.

Question sources

Database Fundamentals practice questions · Question 26

Question 92 | Q01352

The number of entities to which another entity can be associated via a relationship set is expressed as :

A. Entity

B. Cardinality

C. Schema

D. Attributes

Hint

Read the relationship in both directions and check participation constraints.

Answer B: Cardinality

Cardinality measures how often an entity participates and distinguishes one-to-one, one-to-many, and many-to-many. That is the number of possible associations asked about.

Why each choice is right or wrong

A - Incorrect: Entity is a participating object. (PDF pp. 11, 12.)

B - Correct: Cardinality is the association-count constraint. (PDF pp. 11, 12.)

C - Incorrect: Schema describes structure rather than the count. (PDF pp. 11, 12.)

D - Incorrect: Attributes are properties, not the association-count classification. (PDF pp. 11, 12.)

Primary note proof

Study-note section FDB09

Cardinality specifies maximum associated occurrences: 1:1, 1:N or M:N. Minimum cardinality specifies optional/partial participation (zero permitted) or mandatory/total participation (at least one). A maximum of one does not mean participation is compulsory. Relationship degree counts participating entity roles: unary recursive, binary or ternary.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 24-31

Cardinality The number of times an entity of an entity set participates in a relationship set is known as cardinality. Cardinality can be of different types: One to one (1-1) - When each entity in each entity set can take part only once in the relationship, the cardinality is one to one. One to many(1-M)- When entities in one entity set can take part only once in the relationship set and entities in other entity set can take part more than once in the relationship set, cardinality is many to one.

Fundamental of Database Systems module | PDF page 12 | lines 2-7

Many to many (M-M) - When entities in all entity sets can take part more than once in the relationship cardinality is many to many. The participation constraint specifies whether the existence of an entity depends on its being related to another entity via the relationship type. This constraint specifies the minimum number of relationship instances that each entity can participate in and is sometimes called the minimum cardinality constraint. There are two types of participation constraints, total and partial.

Question sources

Database Fundamentals practice questions · Question 27

Question 93 | Q01353

Relations produced from an E-R model will always be in :

A. First normal form

B. Second normal form

C. Third normal form

D. Fourth normal form

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer A: First normal form, assuming correct ER-to-relational mapping.

FDB20 states that properly splitting composite and multivalued ER structures produces ordinary 1NF relations. Nothing in an ER drawing alone proves absence of partial, transitive or multivalued dependencies needed for higher forms. Thus A is the guaranteed level under correct relational mapping; the word always must not be read as excusing an incorrect mapping.

Why each choice is right or wrong

A - Correct: Correctly mapped atomic relations are in 1NF; this is the guaranteed level described here.

B - Incorrect: An ER model alone does not rule out partial dependencies on a composite candidate key.

C - Incorrect: An ER model alone does not establish the full dependency conditions required for 3NF.

D - Incorrect: An ER model alone does not rule out the nontrivial multivalued dependencies addressed by 4NF.

Primary note proof

Study-note section FDB20

> 1NF requires atomic values in the chosen relational domains and no repeating groups in cells. It does not mean merely declaring a primary key. A mapped ER design that correctly splits composite and multivalued structures gives ordinary 1NF tables; ER drawing alone guarantees no higher form.

Study-note section FDB20

> Higher forms inherit 2NF, so no partial dependencies establishes the minimum guaranteed stage only when 1NF is also assumed.

Study-note section FDB20

> A relation whose every candidate key is single-attribute has no nonprime partial-key dependency.

Related reading (context, not primary note proof)

Study-note section FDB21

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 12 | lines 8-24

> 3.4 Mapping ER-models to relational tables 1. For each strong entity type E, create a relation R that includes all the attributes of E. 2. Choose one of the key attributes of E as the primary key for R. For each weak entity type W with owner entity type E (which is a strong entity), create a relation R & include all attributes of ‗W‘ as attributes of R. Also, include the primary key attribute of the relation that correspond to the owner entity type as foreign key attributes of R . 3. For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. Foreign Key approach: Choose one of the relations-say S-and include a foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S. 4. For each binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type. Include the primary key of the relation T as foreign key in S that represents the other entity type participating in R. Include any attributes of the 1:N relation type as attributes of S. 5. For each

n-ary relationship type R, where n>2, create a new relationship S to represent R. Include the primary keys of the relations that represent the participating entity types as foreign key attributes in S. Also include any attributes of the n-ary relationship type as attributes of S.

Fundamental of Database Systems module | PDF page 17 | lines 2-15

First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF). Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Database Fundamentals practice questions · Question 28

Question 94 | Q01354

In ER model the details of the entities are hidden from the user. This process is called :

A. generalization

B. specialization

C. abstraction

D. none of these above

Hint

Trace which schema levels a proposed change affects.

Answer C: abstraction

Abstraction hides unwanted or irrelevant details. Generalization groups subclasses; specialization divides a superclass. Thus hiding entity details describes abstraction.

Why each choice is right or wrong

A - Incorrect: Generalization combines common characteristics, rather than the general detail-hiding process. (PDF pp. 6, 13.)

B - Incorrect: Specialization separates narrower classes. (PDF pp. 6, 13.)

C - Correct: Abstraction is explicitly the process of hiding details. (PDF pp. 6, 13.)

D - Incorrect: The required term is present. (PDF pp. 6, 13.)

Primary note proof

Study-note section FDB04

> The external level gives each user group a selected view. The conceptual level describes the whole logical database: entity types, relationships, data types and constraints. The internal/physical level describes files, record placement and access paths. These are abstraction levels, distinct from client/server deployment tiers.

Study-note section FDB13

> Specialization divides Student into Undergraduate and Graduate; generalization combines common properties into Student.

Study-note section FDB13

> EER extends ER with superclass/subclass structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 6 | lines 2-4

> Data abstraction and Insulation between programs and data- Data Abstraction is a process of hiding unwanted or irrelevant details from the end user. It provides a different view and helps in achieving data independence which is used to enhance the security of data.

Fundamental of Database Systems module | PDF page 13 | lines 9-14

> Specialization is the process of dividing a parent-level entity (super-class) into narrower categories (sub-classes) accordingly to all the possible child categories. Specialization requires the separation of entities based on certain uncommon attributes. The process of extracting common characteristics from two or more classes (sub-classes) and combining them into a generalized superclass is called Generalization. It is the process that does the grouping entities into broader or distributed categories based on certain common attributes.

Question sources

Database Fundamentals practice questions · Question 29

Question 95 | Q01355

The attributes made up of more than one single attributes are called

A. Composite attribute

B. Derived attribute

C. Single value attribute

D. Multi value attribute

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer A: Composite attribute

Composite attributes have smaller meaningful subparts. This differs from multiple values of one attribute and from calculating a value from another.

Why each choice is right or wrong

A - Correct: Composite is the stated divisible structure. (PDF pp. 10, 10.)

B - Incorrect: Derived concerns computation from stored attributes. (PDF pp. 10, 10.)

C - Incorrect: Single-valued concerns count of values, not internal composition. (PDF pp. 10, 10.)

D - Incorrect: Multivalued concerns a set of values rather than components of one value. (PDF pp. 10, 10.)

Primary note proof

Study-note section FDB06

> Simple versus composite asks whether a value has meaningful components: Address may split into City and Street. Single-valued versus multivalued asks how many values one entity may have: a student may have several phone numbers. These dimensions differ: one address may be composite and single-valued. Age derived from birth date is derived, even if also single-valued. Attribute names alone do not establish multiplicity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 15-24

> Composite and Simple (Atomic) Attributes Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings. Attributes that are not divisible are called simple or atomic attributes. Multivalued Attributes Most attributes have a single value for a particular entity; such attributes are called single-valued. For example, Age is a single-valued attribute of a person. In some cases an attribute can have a set of values for the same entity. For instance, a College_degrees attribute for a person. Similarly, one person may not have any college degrees, another person may have one, and a third person may have two or more degrees; therefore, different people can have different numbers of values for the College_degrees attribute. Such attributes are called multivalued.

Fundamental of Database Systems module | PDF page 10 | lines 25-30

> Derived Attributes In some cases, two (or more) attribute values are related. For example, the Age and Birth_date attributes of a person. For a particular person entity, the value of Age can be determined from the current (today‘s) date and the value of that person‘s Birth_date. The Age attribute is hence called a derived attribute and is said to be derivable from the Birth_date attribute, which is called a stored attribute.

Question sources

Database Fundamentals practice questions · Question 30

Question 96 | Q01356

The attribute AGE is calculated from DATE_OF_BIRTH. The attribute AGE is

A. Single valued

B. Multi valued

C. Composite

D. Derived

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer D: Derived

The exact Age/Birth_date example identifies Age as derived. Age can also be single-valued, but the stem asks the classification indicated by calculating it from another attribute; D describes that dependency, while A describes a different axis.

Why each choice is right or wrong

A - Incorrect: Age can be single-valued, but that does not express the calculation relationship asked for. (PDF pp. 10, 10.)

B - Incorrect: No set of ages per person is described. (PDF pp. 10, 10.)

C - Incorrect: No meaningful component attributes of Age are specified. (PDF pp. 10, 10.)

D - Correct: Derived directly matches the module's example and the calculation clause. (PDF pp. 10, 10.)

Primary note proof

Study-note section FDB06

> Simple versus composite asks whether a value has meaningful components: Address may split into City and Street. Single-valued versus multivalued asks how many values one entity may have: a student may have several phone numbers. These dimensions differ: one address may be composite and single-valued. Age derived from birth date is derived, even if also single-valued. Attribute names alone do not establish multiplicity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 19-20

> Most attributes have a single value for a particular entity; such attributes are called single-valued. For example, Age is a single-valued attribute of a person. In some cases an attribute can have a set

Fundamental of Database Systems module | PDF page 10 | lines 25-30

> Derived Attributes In some cases, two (or more) attribute values are related. For example, the Age and Birth_date attributes of a person. For a particular person entity, the value of Age can be determined from the current (today‘s) date and the value of that person‘s Birth_date. The Age attribute is hence called a derived attribute and is said to be derivable from the Birth_date attribute, which is called a stored attribute.

Question sources

Database Fundamentals practice questions · Question 31

Question 97 | Q01357

A student has one official Name and one Date_of_birth, but may have several Phone_number values. Which listed attribute is multivalued for a student?

A. Phone_number

B. Name

C. Date_of_birth

D. All of the mentioned

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer A: A. Phone_number

Multiplicity asks how many values of one attribute may belong to the same entity. The student may have several phone numbers, making Phone_number multivalued. The stated rules allow one official name and one birth date, so those attributes are single-valued. These classifications follow the supplied requirements rather than the names or internal structure of the values.

Why each choice is right or wrong

A - Correct: Several phone values are permitted for one student.

B - Incorrect: The model permits one official name per student.

C - Incorrect: One birth date is permitted per student.

D - Incorrect: Name and birth date are single-valued under the stated rules.

Primary note proof

Study-note section FDB06

Simple versus composite asks whether a value has meaningful components: Address may split into City and Street. Single-valued versus multivalued asks how many values one entity may have: a student may have several phone numbers. These dimensions differ: one address may be composite and single-valued. Age derived from birth date is derived, even if also single-valued. Attribute names alone do not establish multiplicity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 15-24

Composite and Simple (Atomic) Attributes Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings. Attributes that are not divisible are called simple or atomic attributes. Multivalued Attributes Most attributes have a single value for a particular entity; such attributes are called single-valued. For example, Age is a single-valued attribute of a person. In some cases an attribute can have a set of values for the same entity. For instance, a College_degrees attribute for a person. Similarly, one person may not have any college degrees, another person may have one, and a third person may have two or more degrees; therefore, different people can have different numbers of values for the College_degrees attribute. Such attributes are called multivalued.

Question sources

Database Fundamentals practice questions · Question 32

Question 98 | Q01358

For each student, Register_number has one value; Address records all residences and permits several values; SUBJECT_TAKEN permits several subjects; Reference permits several referees. Which listed attribute is single-valued?

A. Register_number

B. Address

C. SUBJECT_TAKEN

D. Reference

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer A: A. Register_number

Single-valued means one value per entity, independently of whether that value can be broken into components. Register_number is single-valued under the stated student rule. Address, SUBJECT_TAKEN and Reference each explicitly permit multiple values for one student. An address can be composite without necessarily being multivalued, but this particular model permits multiple residences.

Why each choice is right or wrong

A - Correct: One register number is assigned per student.

B - Incorrect: Several residence addresses are permitted.

C - Incorrect: Several subjects can be taken by one student.

D - Incorrect: Several referee references are permitted.

Primary note proof

Study-note section FDB06

> Simple versus composite asks whether a value has meaningful components: Address may split into City and Street. Single-valued versus multivalued asks how many values one entity may have: a student may have several phone numbers. These dimensions differ: one address may be composite and single-valued. Age derived from birth date is derived, even if also single-valued. Attribute names alone do not establish multiplicity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 15-24

> Composite and Simple (Atomic) Attributes Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings. Attributes that are not divisible are called simple or atomic attributes. Multivalued Attributes Most attributes have a single value for a particular entity; such attributes are called single-valued. For example, Age is a single-valued attribute of a person. In some cases an attribute can have a set of values for the same entity. For instance, a College_degrees attribute for a person. Similarly, one person may not have any college degrees, another person may have one, and a third person may have two or more degrees; therefore, different people can have different numbers of values for the College_degrees attribute. Such attributes are called multivalued.

Question sources

Database Fundamentals practice questions · Question 33

Question 99 | Q01359

An association of several entities in an Entity-Relation model is called :

A. Tuple

B. Record

C. Relationship

D. Field

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer C: Relationship

A relationship represents an association between entity types. Tuples and records instead represent individual table rows.

Why each choice is right or wrong

A - Incorrect: Tuple is a row. (PDF pp. 11, 13.)

B - Incorrect: Record is row data, not the ER association category. (PDF pp. 11, 13.)

C - Correct: Relationship is the explicit association term. (PDF pp. 11, 13.)

D - Incorrect: Field is an attribute component, not the association. (PDF pp. 11, 13.)

Primary note proof

Study-note section FDB06

> An entity is a distinguishable real or conceptual thing, such as one student or one course. An entity type describes common properties; its current collection of entities is an entity set. A relationship associates entities. In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes. Always inspect the diagram legend.

Study-note section FDB03

> A tuple/row/record contains related attribute values; an attribute/column/field describes a property.

Study-note section FDB03

> A relation is a set of tuples, represented as a table.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 13-15

> A relationship type represents the association between entity types. For example, _Enrolled in‘ is a relationship type that exists between entity Student and Course. In ER diagram, relationship is represented by a diamond and connecting the entities with lines.

Fundamental of Database Systems module | PDF page 13 | lines 24-27

> Tables - In the Relational model, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes. Tuple - It is a single row of a table, which contains a single record.

Question sources

Database Fundamentals practice questions · Question 34

Question 100 | Q01360

To include integrity constraint in an existing relation use:

A. Modify table

B. Drop table

C. Alter table

D. Create table

Hint

Distinguish table definition changes from changes to stored rows.

Answer C: Alter table

Adding an integrity constraint to an existing table changes its definition, so ALTER TABLE is the appropriate command family. It can add a named constraint or a primary key, provided existing data satisfies the rule. DROP TABLE removes the object, and CREATE TABLE defines a new one. MODIFY TABLE does not follow the ALTER TABLE command form required for this structural change. The remaining constraint clause depends on the integrity rule to be enforced.

Why each choice is right or wrong

A - Incorrect: Incorrect: the required command family for this definition change is ALTER TABLE, not MODIFY TABLE.

B - Incorrect: Incorrect: DROP TABLE removes the existing object instead of adding a constraint.

C - Correct: Correct: ALTER TABLE can add a constraint to an existing table definition.

D - Incorrect: Incorrect: CREATE TABLE defines a new table rather than changing this existing one.

Primary note proof

Study-note section FDB30

> ALTER TABLE Employee ADD COLUMN Email VARCHAR(120); adds a column in PostgreSQL-style syntax. ALTER TABLE Employee DROP COLUMN Email; removes it. ALTER TABLE Employee ADD CONSTRAINT uq_email UNIQUE (Email); adds a named constraint. Use ALTER rather than DALTER, and do not separate column_name and column_type with a comma; validate the resulting SQL before execution.

Study-note section FDB26

DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 49 | lines 16-28

Data and Domain Integrity • This is now increasing handled by the database. In Oracle, for example, when defining a table you can specify: • CREATE TABLE table-name (attr2 attr-type NOT NULL, forbids NULL values attrN attr-type CHECK (attrN = UPPER(attrN) verifies that the data meets certain criteria attrO attr-type DEFAULT default_value); Supplies default values Referential Integrity • Ensures that dependent relationships in the data are maintained. In Oracle, for example: • CREATE TABLE table-name (attr1 attr-type PRIMARY KEY, attr2 attr-type NOT NULL, ..., attrM attr-type REFERENCES owner.tablename(attrname) ON DELETE CASCADE, ...

Question sources

Database Fundamentals practice questions · Question 35

Question 101 | Q01361

Which option is not a general SQL integrity-constraint keyword or clause? Distinguish a constraint clause from a business condition that could be expressed inside one.

A. Not null

B. Positive

C. Unique

D. Check predicate

Hint

Read each SQL constraint's scope and enforcement rule.

Answer B: Positive

NOT NULL forbids missing values, UNIQUE enforces uniqueness under the DBMS's NULL rules, and CHECK evaluates a declared predicate. POSITIVE by itself is not a general SQL constraint keyword. A positivity requirement is still a valid business rule and can be written as a CHECK predicate such as CHECK (Amount > 0). The question therefore distinguishes clause syntax from the meaning of a possible rule, making B the unique incorrect keyword rather than rejecting positivity as an integrity requirement.

Why each choice is right or wrong

A - Incorrect: Incorrect selection: NOT NULL is a constraint that forbids NULL values.

B - Correct: Correct selection: POSITIVE is not the standalone constraint keyword; a CHECK predicate can enforce positivity.

C - Incorrect: Incorrect selection: UNIQUE is a constraint enforcing uniqueness.

D - Incorrect: Incorrect selection: CHECK is the clause for enforcing a declared predicate.

Primary note proof

Study-note section FDB16

> Example: CREATE TABLE Enrollment (StudentID INTEGER NOT NULL REFERENCES Student(StudentID), CourseID INTEGER NOT NULL REFERENCES Course(CourseID), Grade INTEGER CHECK (Grade BETWEEN 0 AND 100), PRIMARY KEY (StudentID,CourseID)); Each column declaration gives its name then type, with commas between declarations. CHECK is a constraint clause, not a standalone GRANT-like statement. POSITIVE is not a general SQL constraint keyword; CHECK (Amount > 0) expresses positivity.

Study-note section FDB16

> PRIMARY KEY implies uniqueness and non-NULL. UNIQUE does not generally imply NOT NULL; NULL treatment differs by DBMS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 14 | lines 17-20

> Key Constraints- These are called uniqueness constraints since it ensures that every tuple in the relation should be unique. An attribute that can uniquely identify a tuple in a relation is called the key or primary key of the table. The value of the attribute for different tuples in the relation has to be unique. Primary keys must be unique and cannot be null.

Fundamental of Database Systems module | PDF page 47 | lines 20-27

> • Five types i. Required data ii. attribute domain constraints iii. entity integrity iv. referential integrity v. enterprise constraints i. Required Data • Some attributes must always contain a value -- they cannot have a NULL value

Fundamental of Database Systems module | PDF page 49 | lines 19-22

> • CREATE TABLE table-name (attr2 attr-type NOT NULL, forbids NULL values attrN attr-type CHECK (attrN = UPPER(attrN) verifies that the data meets certain criteria attrO attr-type DEFAULT default_value); Supplies default values

Question sources

Database Fundamentals practice questions · Question 36

Question 102 | Q01362

Foreign key is the one in which the _____ of one relation is referenced in another relation.

A. Foreign key

B. Primary key

C. References

D. Check constraint

Hint

Match the validity requirement to the appropriate constraint definition.

Answer B: Primary key

A foreign key references a matching candidate, usually primary, key in the parent. B is the offered conventional primary-key case; the definition also permits a candidate key, so primary is not an exclusive universal requirement.

Why each choice is right or wrong

A - Incorrect: An arbitrary foreign key is not the required parent unique target. (PDF pp. 48.)

B - Correct: Primary key is the listed usual referenced key. (PDF pp. 48.)

C - Incorrect: References names a mechanism rather than the key category. (PDF pp. 48.)

D - Incorrect: A CHECK predicate is not the referenced key. (PDF pp. 48.)

Primary note proof

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

> iv. Referential Integrity • A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key • Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Question sources

Database Fundamentals practice questions · Question 37

Question 103 | Q01363

Which are uses of data integrity constraints, rather than access authorization? Select all that apply.

A. Control who is allowed access to the data

B. Ensure that duplicate records are not entered into the table

C. Improve the quality of data entered for a specific property (i.e., table column)

D. Prevent users from changing the values stored in the table

Hint

Match the validity requirement to the appropriate constraint definition.

Answer B, C: B. Ensure that duplicate records are not entered into the table; C. Improve the quality of data entered for a specific property (i.e., table column)

Integrity constraints test whether data is valid according to the schema's rules. Uniqueness constraints can reject duplicate keyed records, while domain constraints reject inappropriate values for a column. These support the two selected purposes. Authorization controls who may access the data. Integrity rejects invalid changes rather than prohibiting all changes, so it does not imply that every stored value is immutable.

Why each choice is right or wrong

A - Incorrect: Controlling who may access data is authorization rather than data integrity itself.

B - Correct: Key or uniqueness constraints prevent duplicate keyed records.

C - Correct: Domain constraints validate types and allowed values for a specific column.

D - Incorrect: Integrity permits valid updates; it does not prohibit every change to stored values.

Primary note proof

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 47 | lines 4-19

> ● Security vs Integrity ◦ Database security makes sure that the user is authorised to access information ◦ Database integrity makes sure that (authorised) users use that information correctly ● Integrity constraints ◦ Domain constraints apply to data types ◦ Attribute constraints apply to columns ◦ Relation constraints apply to rows in a single table ◦ Database constraints apply between tables ● Intrarecord integrity (enforcing constraints on contents of fields, etc.) ● Referential Integrity (enforcing the validity of references between records in the database) ●

Concurrency control (ensuring the validity of database updates in a shared multiuser environment) ● The constraints we wish to impose in order to protect the database from becoming inconsistent.

Fundamental of Database Systems module | PDF page 48 | lines 4-12

ii. Attribute domain constraints ● Every attribute has a domain, that is a set of values that are legal for it to use ● For example: ◦ The domain of sex in the employee relation is —M‖ or —F‖ ● Domain ranges can be used to validate input to the database iii. Entity integrity ● The primary key of any entity: ◦ Must be Unique ◦ Cannot be NULL

Question sources

Database Fundamentals practice questions · Question 38

Question 104 | Q01364

The relationship between the two tables are created using _____

A. Candidate Key

B. Primary Key

C. Foreign Key

D. Check Constraint

Hint

Match the validity requirement to the appropriate constraint definition.

Answer C: Foreign Key

A foreign key links child occurrences to the matching parent candidate key. Primary/candidate keys identify the target, but foreign-key referencing creates the intertable link.

Why each choice is right or wrong

A - Incorrect: Candidate keys identify potential reference targets rather than expressing the link alone. (PDF pp. 48.)

B - Incorrect: A primary key identifies a row; a foreign key expresses the reference. (PDF pp. 48.)

C - Correct: The foreign key is explicitly the linking mechanism. (PDF pp. 48.)

D - Incorrect: CHECK restricts values rather than defining this intertable reference. (PDF pp. 48.)

Primary note proof

Study-note section FDB15

Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Study-note section FDB07

A primary key is one selected candidate key; remaining candidates are alternate keys.

Study-note section FDB16

POSITIVE is not a general SQL constraint keyword; CHECK (Amount > 0) expresses positivity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

iv. Referential Integrity • A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key • Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Question sources

Database Fundamentals practice questions · Question 39

Question 105 | Q01365

Customer has candidate keys CustId, Email and ContactNo. CustId is an immutable compact generated identifier; Email and ContactNo may change. A policy prefers stable compact candidate keys. Which is the best primary key under that policy?

A. CustId

B. Email

C. ContactNo

D. CustId and ContactNo

Hint

Test uniqueness and minimality separately when classifying keys.

Answer A: A. CustId

Every candidate key is eligible for primary-key selection, but the policy gives a reason to prefer one. CustId is explicitly stable and compact, while the contact attributes may change. Choosing CustId follows that policy without denying the uniqueness of the other candidates. Combining CustId and ContactNo is nonminimal because CustId alone already identifies each customer.

Why each choice is right or wrong

A - Correct: Its stated stability and compactness satisfy the preference policy.

B - Incorrect: Email is eligible but may change, so it loses under this policy.

C - Incorrect: ContactNo is eligible but may change, so it loses under this policy.

D - Incorrect: Adding ContactNo to an already unique CustId is unnecessary and nonminimal.

Primary note proof

Study-note section FDB07

A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

Prefer stable, compact identifiers when practical, but Email, SSN or EmployeeID is a candidate only if business rules guarantee uniqueness.

Study-note section FDB07

A table can have several candidates and foreign keys but at most one declared primary-key constraint, possibly containing several columns.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Question sources

Database Fundamentals practice questions · Question 40

Question 106 | Q01366

Which of the following statement(s) is/are FALSE about primary key?

A. There can be more than one primary key for a table

B. Columns with string data types cannot be made Primary Key because they are large

C. Both A and B

D. None of the above

Hint

Test uniqueness and minimality separately when classifying keys.

Answer C: Both A and B

The designer selects one candidate key as the primary key; a composite primary key is still one key. No numeric-only restriction appears: a key is defined by uniqueness, not by a prohibition on string type. Thus both A and B are false claims.

Why each choice is right or wrong

A - Incorrect: False: several candidate keys do not mean several primary keys. (PDF pp. 11, 14.)

B - Incorrect: False: the definition imposes uniqueness, not a ban on strings. (PDF pp. 11, 14.)

C - Correct: Both preceding claims conflict with the key definition. (PDF pp. 11, 14.)

D - Incorrect: There are false claims, so none is not correct. (PDF pp. 11, 14.)

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

> A table can have several candidates and foreign keys but at most one declared primary-key constraint, possibly containing several columns.

Study-note section FDB16

> PRIMARY KEY implies uniqueness and non-NULL.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

> Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Fundamental of Database Systems module | PDF page 14 | lines 17-20

> Key Constraints- These are called uniqueness constraints since it ensures that every tuple in the relation should be unique. An attribute that can uniquely identify a tuple in a relation is called the key or primary key of the table. The value of the attribute for different tuples in the relation has to be unique. Primary keys must be unique and cannot be null.

Question sources

Database Fundamentals practice questions · Question 41

Question 107 | Q01367

Which is the subset of SQL commands used to manipulate Oracle Database structures, including tables?

A. Data Definition Language(DDL)

B. Data Manipulation Language(DML)

C. DML and DDL

D. None of the Mentioned

Hint

Classify each SQL command by the kind of change it performs.

Answer A: Data Definition Language(DDL)

Schema-structure manipulation falls under definition language, despite the everyday word manipulate. DML is the separate category for data manipulation.

Why each choice is right or wrong

A - Correct: DDL contains the structural commands. (PDF pp. 26.)

B - Incorrect: DML concerns data operations. (PDF pp. 26.)

C - Incorrect: The asked subset for structures is DDL, not a combined category. (PDF pp. 26.)

D - Incorrect: DDL is offered. (PDF pp. 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 42

Question 108 | Q01368

Using the course's textbook SQL command categories, with RENAME treated as DDL, which listed commands are outside DDL? Select all that apply.

A. RENAME

B. REVOKE

C. GRANT

D. UPDATE

Hint

Classify each SQL command by the kind of change it performs.

Answer B, C, D: B. REVOKE; C. GRANT; D. UPDATE

DDL concerns structural definitions and includes RENAME in this course classification. GRANT and REVOKE manage privileges, placing them in DCL, while UPDATE changes existing data and belongs to DML. All three are consequently outside DDL. Classifying by each statement's effect distinguishes schema operations, access permissions and data changes even though all are expressed using SQL.

Why each choice is right or wrong

A - Incorrect: RENAME is commonly classified as DDL in the note's textbook taxonomy.

B - Correct: REVOKE removes privileges and is DCL rather than DDL.

C - Correct: GRANT assigns privileges and is DCL rather than DDL.

D - Correct: UPDATE changes data and is DML rather than DDL.

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

> 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 54 | lines 6-10

2.2.3 Authorization Rules • Most current DBMS permit the DBA to define —access permissions‖ on a table by table basis (at least) using the GRANT and REVOKE SQL commands • Some systems permit finer grained authorization (most use GRANT and REVOKE on variant views

Question sources

Database Fundamentals practice questions · Question 43

Question 109 | Q01369

Which command empties an existing table while retaining its definition, using DBMS-specific table-emptying mechanisms rather than asserting that rollback is impossible?

A. TRUNCATE command

B. Create command

C. Drop table command

D. Alter table command

Hint

Classify each SQL command by the kind of change it performs.

Answer A: A. TRUNCATE command

TRUNCATE empties a table while retaining the table object. Its implementation, logging and transaction behavior depend on the DBMS, so the answer does not rely on a claim that rollback is impossible. CREATE defines an object, DROP TABLE removes it, and ALTER changes its definition. Only TRUNCATE has the requested table-emptying role among these choices.

Why each choice is right or wrong

A - Correct: TRUNCATE empties the table and keeps its definition.

B - Incorrect: CREATE defines objects rather than emptying an existing table.

C - Incorrect: DROP TABLE removes the definition as well as the table.

D - Incorrect: ALTER changes the table definition rather than serving as the table-emptying command.

Primary note proof

Study-note section FDB26

DELETE removes rows and keeps the table; DROP TABLE removes the table object; ALTER changes its definition. TRUNCATE empties a table using implementation-specific mechanisms. PostgreSQL allows transactional rollback of TRUNCATE, so no rollback information is retained is not a universal SQL property. SQL Server permits DROP DATABASE Sales, suppliers; that comma-list syntax is vendor-specific. SQLite's .databases lists attached names/files in its command-line shell; .database is not the documented command.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

> 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 52 | lines 2-3

> • Part or all of a transaction can be undone using ROLLBACK • COMMIT;

Question sources

Database Fundamentals practice questions · Question 44

Question 110 | Q01370

Which SQL command form removes a database object together with its contained database objects?

A. DROP DATABASE

B. DELETE DATABASE

C. REMOVE DATABASE

D. None of the mentioned

Hint

Classify each SQL command by the kind of change it performs.

Answer A: DROP DATABASE

DROP removes a database object, whereas DELETE removes rows from an existing table. DROP DATABASE is consequently the correct command form for removing a database and its contained objects. DELETE DATABASE does not express that structural operation, and REMOVE DATABASE does not follow the taught DROP syntax. Since the correct DROP DATABASE form is one of the options, the none-of-the-mentioned alternative is also incorrect.

Why each choice is right or wrong

A - Correct: Correct: DROP DATABASE removes the database object and its contained objects.

B - Incorrect: Incorrect: DELETE is row removal; deleting rows does not remove the database object.

C - Incorrect: Incorrect: the database-removal command form uses DROP, not REMOVE.

D - Incorrect: Incorrect: DROP DATABASE is present and performs the requested structural operation.

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Study-note section FDB26

> SQL Server permits DROP DATABASE Sales, suppliers; that comma-list syntax is vendor-specific.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

> 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 45

Question 111 | Q01371

In SQL Server, which statement drops the two databases named Sales and suppliers? Assume permissions and database state allow the operation.

A. DROP DATABASE Sales, NewSales

B. DROP DATABASE Sales, suppliers

C. DROP DATABASE Sales and suppliers

D. DROP DATABASE Sales-suppliers

Hint

Classify each SQL command by the kind of change it performs.

Answer B: B. DROP DATABASE Sales, suppliers

SQL Server permits a comma-separated list of database names after DROP DATABASE. B names exactly Sales and suppliers using that syntax. A names a different second database, while AND and a hyphen do not form the required two-name list. The answer is specific to SQL Server and does not assert that the same multi-database syntax works in every SQL dialect.

Why each choice is right or wrong

A - Incorrect: The second target is NewSales rather than suppliers.

B - Correct: The supported comma list names both required databases.

C - Incorrect: AND is not the separator in this DROP DATABASE syntax.

D - Incorrect: A hyphen does not separate these two database names.

Primary note proof

Study-note section FDB26

DELETE removes rows and keeps the table; DROP TABLE removes the table object; ALTER changes its definition. TRUNCATE empties a table using implementation-specific mechanisms. PostgreSQL allows transactional rollback of TRUNCATE, so no rollback information is retained is not a universal SQL property. SQL Server permits DROP DATABASE Sales, suppliers; that comma-list syntax is vendor-specific. SQLite's .databases lists attached names/files in its command-line shell; .database is not the documented command.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 46

Question 112 | Q01372

Command that is used to add attributes to an existing relation, is said to be

A. Alter

B. Modify

C. Tailor

D. Eliminate

Hint

Distinguish table definition changes from changes to stored rows.

Answer A: ALTER.

Adding attributes changes a relation schema. FDB30 demonstrates ALTER TABLE Employee ADD COLUMN Email VARCHAR(120), keeping the column name next to its type. ALTER is therefore the required command family. The remaining alternatives do not name the general SQL schema-change command shown by the complete note.

Why each choice is right or wrong

A - Correct: ALTER changes an existing relation definition and supports adding a column.

B - Incorrect: MODIFY alone is not the general command for adding attributes to an existing relation.

C - Incorrect: TAILOR is not an SQL schema-change command established here.

D - Incorrect: ELIMINATE is not the SQL command for adding a relation attribute.

Primary note proof

Study-note section FDB30

> ALTER TABLE Employee ADD COLUMN Email VARCHAR(120); adds a column in PostgreSQL-style syntax. ALTER TABLE Employee DROP COLUMN Email; removes it. ALTER TABLE Employee ADD CONSTRAINT uq_email UNIQUE (Email); adds a named constraint. Use ALTER rather than DALTER, and do not separate column_name and column_type with a comma; validate the resulting SQL before execution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

> 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 47

Question 113 | Q01373

Which keyword pair changes a table definition to remove a specified column, rather than changing or deleting row values? Treat the comma in an option as separating keyword names.

A. UPDATE TABLE

B. TRUNCATE COLUMN

C. ALTER, DROP

D. DELETE COLUMN

Hint

Distinguish table definition changes from changes to stored rows.

Answer C: ALTER, DROP

Removing a column changes the table's definition. The appropriate form is ALTER TABLE followed by DROP COLUMN, so the keyword pair represented by C is correct. Its comma separates the two keyword names rather than belonging in executable SQL. UPDATE changes stored row values, DELETE removes rows, and TRUNCATE empties a table while retaining its definition; none of those operations removes the specified column definition.

Why each choice is right or wrong

A - Incorrect: Incorrect: UPDATE changes row values rather than removing a column definition.

B - Incorrect: Incorrect: TRUNCATE empties a table; TRUNCATE COLUMN is not the column-removal form.

C - Correct: Correct: ALTER TABLE ... DROP COLUMN removes a column definition.

D - Incorrect: Incorrect: DELETE removes rows, not a column from the schema.

Primary note proof

Study-note section FDB30

> ALTER TABLE Employee ADD COLUMN Email VARCHAR(120); adds a column in PostgreSQL-style syntax. ALTER TABLE Employee DROP COLUMN Email; removes it. ALTER TABLE Employee ADD CONSTRAINT uq_email UNIQUE (Email); adds a named constraint. Use ALTER rather than DALTER, and do not separate column_name and column_type with a comma; validate the resulting SQL before execution.

Study-note section FDB26

> DELETE removes rows and keeps the table; DROP TABLE removes the table object; ALTER changes its definition.

Study-note section FDB26

> TRUNCATE empties a table using implementation-specific mechanisms.

Study-note section FDB29

UPDATE Book SET Price = 600 WHERE ID = 'B001'; changes matching rows. UPDATE without WHERE targets every row.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 48

Question 114 | Q01374

Which one of these is used with SELECT clause to fetch all columns from a table?

A. ALL

B. *

C. DISTINCT

D. AS

Hint

Trace how SELECT forms its result columns and rows.

Answer B: The asterisk (*).

FDB27 explicitly says SELECT * selects all columns. ALL and DISTINCT instead govern duplicate result rows, so neither means every column. AS introduces an alias. A query still needs the appropriate FROM source for table columns; the question asks only which SELECT-list symbol requests them all.

Why each choice is right or wrong

A - Incorrect: ALL permits duplicate result rows and does not expand the result to every column.

B - Correct: The wildcard * requests every column from the selected table source.

C - Incorrect: DISTINCT removes duplicate result rows rather than selecting all available columns.

D - Incorrect: AS supplies an alias for a result expression or source, not a wildcard column list.

Primary note proof

Study-note section FDB27

SELECT Name,Address FROM Customer; produces only those mailing-label fields. SELECT * selects all columns; DISTINCT removes duplicate result rows; AS supplies an alias; ALL permits duplicates and does not mean all columns. SQL normally preserves duplicates without DISTINCT.

A query reading table columns needs an appropriate FROM source. A literal-only query such as SELECT 1 may legally omit FROM.

Study-note section FDB27

WHERE alone is a clause rather than a complete read-only statement, so it cannot be compared as an equivalent executable command.

Study-note section FDB27

Typical written clause order is SELECT … FROM … WHERE … GROUP BY … HAVING … ORDER BY ….

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 27 | lines 16-21

Example: Identify all managers who work in a London branch SELECT * FROM Staff s, Branch b WHERE s.branchNo = b.branchNo AND s.position = _Manager‘ AND b.city = _london‘;

Fundamental of Database Systems module | PDF page 52 | lines 4-6

● SET TRANSACTION READ ONLY; ● SELECT NAME, ADDRESS FROM WORKERS; ● SELECT MANAGER, ADDRESS FROM PLACES;

Question sources

Database Fundamentals practice questions · Question 49

Question 115 | Q01375

A parent table has a DATE column, a nullable data column and a primary-key column referenced by existing child rows with ON UPDATE RESTRICT. Which column's change to a different otherwise valid value is blocked specifically because the old key is still referenced? No other constraints or triggers block changes.

A. DATE type columns in the table

B. Columns which allows NULL values in the table

C. A primary key column which also serves as foreign key reference in another table

D. All of the above

Hint

Read each SQL constraint's scope and enforcement rule.

Answer C: C. A primary key column which also serves as foreign key reference in another table

The stated restrictive action prevents changing the parent key while existing children still refer to its old value. This is a consequence of the particular reference state and action, rather than inherent immutability of every primary key. DATE and nullable columns can accept otherwise valid changes. Therefore only the referenced primary-key column has the specified update block.

Why each choice is right or wrong

A - Incorrect: DATE values can be changed to other permitted dates.

B - Incorrect: Allowing NULL does not itself block valid updates.

C - Correct: Existing child references plus ON UPDATE RESTRICT block this key change.

D - Incorrect: The date and nullable columns are not blocked under the given conditions.

Primary note proof

Study-note section FDB16

> PRIMARY KEY implies uniqueness and non-NULL. UNIQUE does not generally imply NOT NULL; NULL treatment differs by DBMS. A CHECK can pass when its result is unknown, so add NOT NULL if missing values must be forbidden. Parent updates/deletes may use NO ACTION/RESTRICT, CASCADE or SET NULL. A referenced key is not inherently unchangeable: legality depends on matching children and the declared action.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

> iv. Referential Integrity ● A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key ● Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Fundamental of Database Systems module | PDF page 49 | lines 4-11

> Deletion rules ● A row should not be deleted from the referenced table (parent) if there are matching rows in the referencing table (child) ● Three ways to handle this ◦ Restrict -- disallow the delete ◦ Nullify -- reset the foreign keys in the child to some NULL or dummy value ◦ Cascade -- Delete all rows in the child where there is a foreign key matching the key in the parent row being deleted

Fundamental of Database Systems module | PDF page 26 | lines 10-11

> 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 50

Question 116 | Q01376

Using the course classification that includes SELECT in DML, which listed language terms can describe application requests to retrieve database data? Select all that apply.

A. DML

B. DDL

C. Query language

D. All of the Mentioned

Hint

Classify each SQL command by the kind of change it performs.

Answer A, C: A. DML; C. Query language

A query language expresses requests for data. The course also includes SELECT retrieval within DML, alongside insertion, update and deletion. Both terms can therefore describe the application's retrieval request at different levels of terminology. DDL changes structural definitions instead of expressing the requested retrieval. Since DDL does not fit, the aggregate all-of-the-mentioned choice is also false.

Why each choice is right or wrong

A - Correct: DML includes retrieval with SELECT in the note's classification.

B - Incorrect: DDL changes schema structure rather than expressing ordinary data retrieval requests.

C - Correct: A query language expresses requests for data, directly matching the stem.

D - Incorrect: DDL does not fit the retrieval role, so the inclusive choice is false.

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 24 | lines 5-9

> Relational algebra is the basic set of operations for the relational model. These operations enable a user to specify basic retrieval requests (or queries). The result of an operation is a new relation, which may have been formed from one or more input relations. The algebra operations thus produce new relations. Since each operation returns a relation, operations can be composed (Algebra is —closed‖.)

Question sources

Database Fundamentals practice questions · Question 51

Question 117 | Q01377

Which of the following is not included in DML (Data Manipulation Language)?

A. Insert

B. Update

C. Delete

D. Alter

Hint

Classify each SQL command by the kind of change it performs.

Answer D: Alter

INSERT, UPDATE and DELETE are all in the DML subsection. ALTER is instead in the preceding DDL list.

Why each choice is right or wrong

A - Incorrect: INSERT is listed as DML. (PDF pp. 26.)

B - Incorrect: UPDATE is listed as DML. (PDF pp. 26.)

C - Incorrect: DELETE is listed as DML. (PDF pp. 26.)

D - Correct: ALTER is the DDL exception. (PDF pp. 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

> 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 52

Question 118 | Q01378

In the course's SQL command taxonomy, how is TRUNCATE classified?

A. DDL

B. DML

C. DCL

D. TCL

Hint

Classify each SQL command by the kind of change it performs.

Answer A: DDL

The course groups TRUNCATE with DDL commands. It empties a table through mechanisms that depend on the implementation, but that fact does not turn it into a privilege-management or transaction-boundary command. DML covers commands such as INSERT, UPDATE and DELETE; DCL covers GRANT and REVOKE; TCL covers COMMIT, ROLLBACK and SAVEPOINT. The requested classification is the course convention, not a claim that TRUNCATE has identical recovery behavior in every DBMS.

Why each choice is right or wrong

A - Correct: Correct: the course's textbook classification includes TRUNCATE under DDL.

B - Incorrect: Incorrect: DML includes row operations such as INSERT, UPDATE and DELETE; TRUNCATE is assigned to DDL in this taxonomy.

C - Incorrect: Incorrect: DCL manages privileges through commands such as GRANT and REVOKE.

D - Incorrect: Incorrect: TCL controls transaction boundaries or rollback points, unlike TRUNCATE.

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

> 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 53

Question 119 | Q01379

A type of query that is placed within a WHERE or HAVING clause of another query is called

A. Master query

B. Sub query

C. Super query

D. Multi-query

Hint

Distinguish a stored query definition from stored data and trace subquery scope.

Answer B: Sub query

The optimization section identifies nested queries as separate query blocks, and the view example places a SELECT inside an outer WHERE. Subquery is the offered name matching this inner/nested query; WHERE or HAVING placement does not define every possible subquery location.

Why each choice is right or wrong

A - Incorrect: Master describes neither the inner query nor the module's nesting distinction. (PDF pp. 34, 58.)

B - Correct: Subquery corresponds to the nested inner SELECT demonstrated. (PDF pp. 34, 58.)

C - Incorrect: Super query does not name the inner block. (PDF pp. 34, 58.)

D - Incorrect: Multi-query does not identify the nesting relationship. (PDF pp. 34, 58.)

Primary note proof

Study-note section FDB33

> A subquery is a SELECT nested inside another statement. It may appear in WHERE, HAVING or FROM, not only the first two. IN tests membership; EXISTS tests whether at least one qualifying row exists. Correlated subqueries refer to the outer row. A scalar subquery must produce an appropriate single value. Beware NOT IN when its subquery can return NULL; NOT EXISTS often expresses the intended absence more safely.

Study-note section FDB33

> CREATE VIEW CSStudent AS SELECT StudentID,Name FROM Student WHERE Major = 'CS'; stores a query definition and presents a virtual table.

Study-note section FDB33

> An ordinary view does not necessarily store its result; a materialized view does.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 34 | lines 12-16

> • Query block: The basic unit that can be translated into the algebraic operators and optimized. • A query block contains a single SELECT-FROM-WHERE expression, as well as GROUP BY and HAVING clause if these are part of the block. • Nested queries within a query are identified as separate query blocks.

Fundamental of Database Systems module | PDF page 58 | lines 15-20

> View example CREATE VIEW OwnDept AS SELECT Name, Phone FROM Employee WHERE Department = (SELECT Department FROM Employee WHERE name = USER)

Question sources

Database Fundamentals practice questions · Question 54

Question 120 | Q01380

A data manipulation command the combines the records from one or more tables is called

A. SELECT

B. PROJECT

C. JOIN

D. PRODUCT

Hint

Compute matching pairs, then account for any preserved unmatched rows.

Answer C: JOIN

The DML list contains JOIN, and the query-plan example describes joining Staff and Branch into a joined relation. This directly matches combining records rather than selecting columns alone.

Why each choice is right or wrong

A - Incorrect: SELECT can retrieve data, but JOIN names the combining operation illustrated. (PDF pp. 26, 28.)

B - Incorrect: PROJECT selects columns rather than linking records by the illustrated condition. (PDF pp. 26, 28.)

C - Correct: JOIN is explicitly used to combine the example relations. (PDF pp. 26, 28.)

D - Incorrect: PRODUCT combines all pairs; it is the broader Cartesian operation rather than the illustrated join command. (PDF pp. 26, 28.)

Primary note proof

Study-note section FDB31

An inner join returns matching pairs satisfying ON. An equijoin uses equality; a theta join permits another comparison. A natural join is ordinarily an inner equality join on every same-named column and merges those duplicate columns. JOIN ... USING (DeptID) specifies shared join columns explicitly. CROSS JOIN forms every pair without a matching condition.

Study-note section FDB35

Rename ρ changes relation or attribute names.

Study-note section FDB35

Identify the join variant before predicting its behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 10-11

7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 28 | lines 17-24

Query 2 (Better) (position=„Manager“)^(city=„London“) (Staff Staff.branchNo = Branch.branchNo Branch) - Again requires (1000+50) disk accesses to read from Staff and Branch - Joins Staff and Branch on branchNo with 1000 tuples (1 employee : 1 branch) - Requires (1000) disk access to read in joined relation and check predicate Total Work = (1000+50) + 2*(1000) = 3050 I/O operations

Question sources

Database Fundamentals practice questions · Question 55

Question 121 | Q01382

Data manipulation language (DML) includes statements that modify the _____ of a database.

A. Structure

B. Data

C. User

D. Size

Hint

Classify each SQL command by the kind of change it performs.

Answer B: Data

Data-definition commands are distinct from data-manipulation commands such as INSERT, DELETE and UPDATE. Their category concerns the stored data, while SELECT is also included as a retrieval command that need not modify state.

Why each choice is right or wrong

A - Incorrect: Structure belongs to definition language. (PDF pp. 26.)

B - Correct: Data is the manipulated content. (PDF pp. 26.)

C - Incorrect: User administration is not the target category here. (PDF pp. 26.)

D - Incorrect: Size can change as a consequence, not the category manipulated. (PDF pp. 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 57

Question 122 | Q01383

Which listed term names the standard SQL expression form for converting a value to a requested data type, rather than vendor-specific conversion syntax?

A. Cast

B. Drop type

C. Alter type

D. Convert

Hint

Check set membership, duplicates and compatible result types.

Answer A: A. Cast

CAST is the standard expression form that requests conversion of a value to a target type. CONVERT can also perform conversions in supporting vendor dialects, but the question specifically excludes vendor-specific syntax. DROP TYPE and ALTER TYPE act on type definitions rather than converting one expression value. The target value must still be representable in the requested type.

Why each choice is right or wrong

A - Correct: CAST supplies the requested standard expression conversion.

B - Incorrect: DROP TYPE removes a type definition.

C - Incorrect: ALTER TYPE changes a type definition.

D - Incorrect: CONVERT is vendor-specific in the supplied course distinction.

Primary note proof

Study-note section FDB34

> CAST('42' AS INTEGER) converts an expression to a requested type if the value permits it. CONVERT is vendor-specific and can also perform type conversion; specify the SQL dialect when comparing it with CAST. In SELECT xyz FROM table1 UNION xyz FROM table2, the second SELECT is missing. In the corrected query, xyz is a selected column/expression, not a table name.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 19 | lines 11-25

> 5.2 Choosing Attribute Data Types CHAR - fixed-length character VARCHAR() - variable-length characters DATE - actual date BLOB - binary large object (good for graphics images, audio, sound clips, etc.) NUMERIC - positive/negative numbers Exact • SMALLINT → 16 bit • INT → 32 bit • LONG → 64 bit • DECIMAL → 128 bit ... Approximate • FLOAT (single precision) → 32 bit with ≈ 6-9 digit precision... • REAL(variation of double) → 32 bit with ≈ 15 digit precision... • DOUBLE (double precision) → 64 bit with ≈ 32 bit precision...

Fundamental of Database Systems module | PDF page 26 | lines 4-11

SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 58

Question 123 | Q01384

Which SQL keyword combines the rows returned by two compatible SELECT statements into one result set, removing duplicates by default?

A. COMBINE

B. CONCAT

C. JOIN

D. UNION

Hint

Check set membership, duplicates and compatible result types.

Answer D: UNION

UNION combines the rows of compatible query results and removes duplicates by default. It is written between the SELECT statements. JOIN instead combines input rows through matching or product behavior within a query, rather than taking the set union of completed results. CONCAT concerns concatenation of values, and COMBINE does not form the required SQL set-operation syntax. D is therefore the unique correct selection for this result-set operation.

Why each choice is right or wrong

A - Incorrect: Incorrect: COMBINE is not the UNION set-operation keyword required between these SELECT statements.

B - Incorrect: Incorrect: concatenating values does not perform the requested union of query-result rows.

C - Incorrect: Incorrect: JOIN forms combinations of input rows rather than this union of compatible SELECT results.

D - Correct: Correct: UNION combines compatible query results and removes duplicates by default.

Primary note proof

Study-note section FDB34

SELECT ID FROM A UNION SELECT ID FROM B; combines compatible result columns and removes duplicate rows by default. UNION ALL retains duplicates. Explicit UNION DISTINCT is accepted in some dialects, including PostgreSQL; it is not a separate algebra rule. INTERSECT returns common rows; EXCEPT returns first-result rows absent from the second. Inputs need the same column count and compatible corresponding types.

Related reading (context, not primary note proof)

Study-note section FDB31

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Fundamental of Database Systems module | PDF page 26 | lines 4-11

SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 59

Question 124 | Q01385

What is xyz in the following SQL statement? SELECT xyz FROM table1 UNION xyz FROM table2;

A. row name

B. column name

C. table name

D. database name

Hint

Check set membership, duplicates and compatible result types.

Answer B: xyz denotes a selected column/expression; the second SELECT is missing.

The printed statement is malformed because the second branch says UNION xyz rather than UNION SELECT xyz. Nevertheless, the role of xyz in the first SELECT list is clear: it is a column or expression, not a row, table or database name. FDB34 explicitly identifies that role and the required repair, so the terminology question remains answerable.

Why each choice is right or wrong

A - Incorrect: SQL does not select a row by placing its row name in this SELECT-list position.

B - Correct: xyz occupies the selected-column position; the second branch needs its missing SELECT keyword.

C - Incorrect: The table sources are table1 and table2, which follow FROM rather than SELECT.

D - Incorrect: No database identifier occupies xyz's SELECT-list position in this statement.

Primary note proof

Study-note section FDB34

> SELECT ID FROM A UNION SELECT ID FROM B; combines compatible result columns and removes duplicate rows by default. UNION ALL retains duplicates. Explicit UNION DISTINCT is accepted in some dialects, including PostgreSQL; it is not a separate algebra rule. INTERSECT returns common rows; EXCEPT returns first-result rows absent from the second. Inputs need the same column count and compatible corresponding types.

Study-note section FDB34

> In the corrected query, xyz is a selected column/expression, not a table name.

Study-note section FDB34

> In SELECT xyz FROM table1 UNION xyz FROM table2, the second SELECT is missing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

> Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations. Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 60

Question 125 | Q01386

In a dialect that accepts explicit UNION DISTINCT, which listed modifier requests duplicate elimination when used with UNION?

A. ALL

B. NARROW

C. STRICT

D. DISTINCT

Hint

Check set membership, duplicates and compatible result types.

Answer D: DISTINCT

UNION eliminates duplicate result rows by default. In the stated dialect, DISTINCT explicitly requests that same behavior, while ALL retains duplicates. NARROW and STRICT are not modifiers in the UNION [ALL | DISTINCT] form being tested. This classification does not imply that every DBMS accepts the explicit DISTINCT spelling: the question states that support, and bare UNION remains sufficient to request duplicate elimination.

Why each choice is right or wrong

A - Incorrect: Incorrect: ALL retains duplicates, the opposite of the requested behavior.

B - Incorrect: Incorrect: NARROW is not an ALL or DISTINCT modifier in the stated UNION form.

C - Incorrect: Incorrect: STRICT is not an ALL or DISTINCT modifier in the stated UNION form.

D - Correct: Correct: DISTINCT explicitly requests duplicate elimination in the stated dialect.

Primary note proof

Study-note section FDB34

> SELECT ID FROM A UNION SELECT ID FROM B; combines compatible result columns and removes duplicate rows by default. UNION ALL retains duplicates. Explicit UNION DISTINCT is accepted in some dialects, including PostgreSQL; it is not a separate algebra rule. INTERSECT returns common rows; EXCEPT returns first-result rows absent from the second. Inputs need the same column count and compatible corresponding types.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-16

> Basic operations: Selection (σ) - Selects a subset of rows from relation. Projection (π) - Deletes unwanted columns from relation. Cross-product (x) - Allows us to combine two relations.

Set-difference (-) - Tuples in relation 1, but not in relation 2. Union (U) - Tuples in relation 1 and in relation 2. Intersection (∩) - Only tuples in relation 1 and in relation 2 (only the common tuples).

Fundamental of Database Systems module | PDF page 26 | lines 4-11

SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 61

Question 126 | Q01387

The database design prevents some data from being stored due to _____.

A. Deletion anomalies

B. Insertion anomalies

C. Update anomalies

D. Selection anomalies

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer B: Insertion anomalies

Insertion anomaly is explicitly the inability to insert a new record properly when the whole primary key is unknown. The other anomalies concern losing information on deletion or inconsistent/multiple updates.

Why each choice is right or wrong

A - Incorrect: Deletion anomaly loses existing related information. (PDF pp. 16.)

B - Correct: Insertion anomaly prevents a proper new insertion. (PDF pp. 16.)

C - Incorrect: Update anomaly concerns changing existing values. (PDF pp. 16.)

D - Incorrect: Selection anomaly is not among the module's stated three types. (PDF pp. 16.)

Primary note proof

Study-note section FDB19

Normalization separates facts according to their determinants, reducing avoidable repetition and associated anomalies. It does not require seeing bad current data: declared dependencies suffice. Adding a generated row ID may make insertion technically possible, but does not remove

repeated facts or fix the dependency design. Reduced redundancy can improve integrity and storage use, but extra keys and joins mean neither storage savings nor faster retrieval is guaranteed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 5-16

4.3 Database Anomalies Database Anomalies are the flaws in databases which occurs because of poor database design and storing everything in a single flat database. They are problems that are caused usually by redundant data. Types of database anomalies are Insertion anomaly, Deletion anomaly, and Update anomaly. Insertion Anomaly - occurs when the entire primary key is not known and the database cannot insert a new record properly. This would violate entity integrity. It can be avoided by using a sequence number for the primary key. Deletion Anomaly - happens when a record is deleted that results in the loss of other related data. It can be avoided by normalizing tables in a database to minimize their dependency. Update Anomaly - occurs when a change to one attribute value causes the database to either contain inconsistent data or causes multiple records to need changing. It may be prevented by making sure tables are in third normal form.

Question sources

Database Fundamentals practice questions · Question 62

Question 127 | Q01388

In the course terminology, which labels can name a drawing of relational tables and their primary-key/foreign-key references? Select all that apply. ER Diagram means a conceptual entity/relationship drawing; Relational Schema means a textual table definition here.

A. ER Diagram

B. Database Diagram

C. Schema Diagram

D. Relational Schema

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer B, C: B. Database Diagram; C. Schema Diagram

A schema diagram draws relational tables and their key references. Database diagram is also used for that depiction in the course terminology, so both labels qualify. The question explicitly distinguishes a conceptual ER drawing and a textual relational schema from the requested table-reference drawing. This makes the two accepted names compatible without treating every representation of a database as the same artifact.

Why each choice is right or wrong

A - Incorrect: A conceptual ER diagram represents entities and relationships rather than specifically depicting mapped table key references.

B - Correct: Database diagram can denote the table-and-key depiction also called a schema diagram.

C - Correct: A schema diagram explicitly draws tables and their primary/foreign-key references.

D - Incorrect: A relational schema defines structures and constraints but need not itself be a diagram.

Primary note proof

Study-note section FDB03

> A schema describes names, structures and constraints; an instance, state or snapshot is the data at a particular time. Inserting a student changes the state, while adding a column changes the schema. Metadata is data describing data. The catalog or data dictionary records table definitions, types, constraints and often access privileges and optimizer statistics. A schema diagram draws tables and key references; labels such as database diagram can overlap.

Study-note section FDB06

> An entity is a distinguishable real or conceptual thing, such as one student or one course.

Study-note section FDB06

> In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 27-31

> 3.3 ERD (Entity Relationship Diagram) ERD a graphical technique for representing a database schema. It is a modeling tool used to depict graphically a DB design before it is actually implemented. It shows the various entities being modeled and the important relationships among them. ERD has three basic components Entity, Relationship and Attribute.

Fundamental of Database Systems module | PDF page 14 | lines 2-3

> Relation Schema: A relation schema represents the name of the relation with its attributes (columns).

Fundamental of Database Systems module | PDF page 48 | lines 13-17

> iv. Referential Integrity ● A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key ● Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Question sources

Database Fundamentals practice questions · Question 63

Question 128 | Q01389

R(A,B,C) has atomic attributes and exactly the dependencies implied by A→B and B→C. A is the sole candidate key; B is not a superkey and C is nonprime. Moving upward through 1NF, 2NF, 3NF and BCNF, which listed normal form is the first that R fails?

A. BCNF

B. 1NF

C. 2NF

D. 3NF

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer D: D. 3NF

The atomic relation satisfies 1NF. Its only candidate key is the single attribute A, so no nonprime dependency can involve a proper part of that key, and 2NF holds. However, B→C has a determinant that is not a superkey and a dependent that is nonprime. This violates the stated 3NF test, so 3NF is the listed form it fails.

Why each choice is right or wrong

A - Incorrect: BCNF also fails, but 3NF is the earlier failed level in the stated hierarchy.

B - Incorrect: Atomic values satisfy 1NF.

C - Incorrect: A single-attribute candidate key has no nonprime partial-key dependency.

D - Correct: B→C violates 3NF because B is not a superkey and C is nonprime.

Primary note proof

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key. For Enrollment(SID,CID,SName,CTitle,Grade), assume key (SID,CID), SID→SName and CID→CTitle. Split into Student(SID,SName), Course(CID,CTitle) and Enrollment(SID,CID,Grade). Grade still depends on the whole pair. A relation whose every candidate key is single-attribute has no nonprime partial-key dependency. Higher forms inherit 2NF, so no partial dependencies establishes the minimum guaranteed stage only when 1NF is also assumed.

Study-note section FDB17

> Full dependency requires all determinant attributes; partial dependency holds through a proper subset.

Study-note section FDB21

With atomic values the highest form is 1NF, even though a transitive dependency also exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 15 | lines 25-28

Transitive Dependency - In mathematics and logic, a transitive relationship is a relationship of the following form: —If A → B, and if also B → C, then A → C.‖ Example: If X >Y and Y >Z then X >Z. Generalized way of describing transitive dependency is that:

Fundamental of Database Systems module | PDF page 16 | lines 2-4

If A functionally governs B, AND If B functionally governs C THEN _A‘ functionally governs _C‘, provided that neither C nor B determines A. A functional dependency is said to be transitive if it is indirectly formed by two functional dependencies.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Database Fundamentals practice questions · Question 64

Question 129 | Q01390

Consider the relation Sale(Date, Customer, Product, Vendor, VendorCity, SalesRep){Date, Customer, Product} is the composite candidate key and the following functional dependencies are also given:Vendor => VendorCity, Product => VendorWhat is the highest normal form of the sale relation?

A. 0NF

B. 1NF

C. 2NF

D. 3NF

Hint

Test determinants and prime attributes against the stated normal-form rules.

Answer B: 1NF

Assuming atomic values as customary for a relation, Product is a proper subset of the stated composite key and determines non-prime Vendor. Vendor→VendorCity adds another non-key dependency. Product→Vendor alone violates 2NF, so the highest supported form is 1NF. No stated dependency makes these non-prime attributes alternative keys.

Why each choice is right or wrong

A - Incorrect: 0NF is not warranted when attributes are atomic. (PDF pp. 15, 17.)

B - Correct: 1NF is the highest form under the explicit atomicity assumption. (PDF pp. 15, 17.)

C - Incorrect: 2NF fails because Vendor depends on Product alone. (PDF pp. 15, 17.)

D - Incorrect: 3NF requires 2NF, which already fails. (PDF pp. 15, 17.)

Primary note proof

Study-note section FDB21

In Sale(Date,Customer,Product,Vendor,VendorCity,SalesRep), assume key (Date,Customer,Product), Product→Vendor and Vendor→VendorCity. Product is a proper subset of the key, so Vendor and VendorCity violate 2NF. With atomic values the highest form is 1NF, even though a transitive dependency also exists. 4NF addresses nontrivial multivalued dependencies with nonsuperkey determinants; 5NF addresses join dependencies. Do not confuse a multivalued dependency with a list stored inside a cell.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 15 | lines 11-17

Partial Dependency - If an attribute which is not a member of the key is dependent on some part of the key (if we have composite/candidate key) then that attribute is partially functionally dependent on the key. Let {A, B} is the key set and C is no key attribute, then if {A, B} → C and [B → C or A → C] holds. We say C is partially functionally dependent on {A, B}. Partial dependency occurs when a non-prime attribute is functionally dependent on part of a candidate/composite key.

Fundamental of Database Systems module | PDF page 17 | lines 2-15

First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF). Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Question sources

Database Fundamentals practice questions · Question 65

Question 130 | Q01391

Third normal form is based on the concept of _____

A. Closure Dependency

B. Transitive Dependency

C. Normal Dependency

D. Functional Dependency

Hint

Test determinants and prime attributes against the stated normal-form rules.

Answer B: Transitive Dependency

The defining extra condition of 3NF over 2NF is non-transitive dependence of non-prime attributes on keys. Functional dependency is the broader framework, but transitive dependency is the particular concept distinguishing this stage.

Why each choice is right or wrong

A - Incorrect: Closure dependency is not the stated criterion. (PDF pp. 18.)

B - Correct: Transitive dependency is the specific concept addressed. (PDF pp. 18.)

C - Incorrect: Normal dependency is not a defined dependency category. (PDF pp. 18.)

D - Incorrect: Functional dependency is broader and does not identify the distinctive 3NF condition. (PDF pp. 18.)

Primary note proof

Study-note section FDB21

> For every nontrivial FD X→A, 3NF permits it when X is a superkey or A is prime. BCNF requires X to be a superkey. The mnemonic whole key and nothing but the key describes the common single-key case; it cannot replace the full tests when keys overlap. Remove dependencies of nonprime facts on nonkey determinants.

Study-note section FDB21

> With atomic values the highest form is 1NF, even though a transitive dependency also exists.

Study-note section FDB21

> Do not confuse a multivalued dependency with a list stored inside a cell.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

> Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Database Fundamentals practice questions · Question 66

Question 131 | Q01392

Which of the following statements does not change the state of the database?

A. Where

B. Select

C. Insert

D. Drop

Hint

Trace how SELECT forms its result columns and rows.

Answer B: Select

The read-only transaction example uses SELECT to retrieve rows from two tables. SELECT is the complete retrieval statement among these choices; WHERE is only a clause and not a standalone SQL statement. The question is read as asking about valid statements.

Why each choice is right or wrong

A - Incorrect: WHERE alone is not a complete statement, so it cannot be the requested statement. (PDF pp. 52, 26.)

B - Correct: SELECT is demonstrated in a read-only transaction. (PDF pp. 52, 26.)

C - Incorrect: INSERT is a manipulation command, not the demonstrated read-only request. (PDF pp. 52, 26.)

D - Incorrect: DROP is a structural DDL command, not the demonstrated retrieval. (PDF pp. 52, 26.)

Primary note proof

Study-note section FDB27

> SELECT Name,Address FROM Customer; produces only those mailing-label fields. SELECT * selects all columns; DISTINCT removes duplicate result rows; AS supplies an alias; ALL permits duplicates and does not mean all columns. SQL normally preserves duplicates without DISTINCT. A query reading table columns needs an appropriate FROM source. A literal-only query such as SELECT 1 may legally omit FROM.

Study-note section FDB27

> WHERE alone is a clause rather than a complete read-only statement, so it cannot be compared as an equivalent executable command.

Study-note section FDB27

Typical written clause order is SELECT ... FROM ... WHERE ... GROUP BY ... HAVING ... ORDER BY ....

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 52 | lines 4-11

• SET TRANSACTION READ ONLY; • SELECT NAME, ADDRESS FROM WORKERS; • SELECT MANAGER, ADDRESS FROM PLACES; • COMMIT; • Freezes the data for the user in both tables before either select retrieves any rows, so that changes that occur concurrently will not show up • Commits before and after ensure any uncompleted transactions are finish, and then release the frozen data when done

Fundamental of Database Systems module | PDF page 26 | lines 7-11

7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Database Fundamentals practice questions · Question 67

Question 132 | Q01500

A relation is in 1NF and has no nonprime attribute partially dependent on any candidate key. What is the strongest normal form guaranteed by these facts alone?

A. First

B. BCNF

C. Third

D. Second

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer D: D. Second

The two stated conditions are exactly the requirements for 2NF: 1NF plus the absence of nonprime partial dependencies on candidate keys. First normal form is also true but is a weaker guarantee. Third normal form and BCNF need additional conditions on other functional dependencies. The relation might satisfy those higher forms, but the given facts alone establish only 2NF.

Why each choice is right or wrong

A - Incorrect: 1NF is given, but 2NF is the stronger guaranteed form.

B - Incorrect: BCNF additionally requires all nontrivial determinants to be superkeys.

C - Incorrect: 3NF needs further checks on determinants and prime dependents.

D - Correct: The conditions directly establish 2NF as the strongest guaranteed form.

Primary note proof

Study-note section FDB20

> 1NF requires atomic values in the chosen relational domains and no repeating groups in cells. It does not mean merely declaring a primary key. A mapped ER design that correctly splits composite and multivalued structures gives ordinary 1NF tables; ER drawing alone guarantees no higher form.

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Study-note section FDB20

> A relation whose every candidate key is single-attribute has no nonprime partial-key dependency.

Related reading (context, not primary note proof)

Study-note section FDB21

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 2-15

> First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF). Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

> Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of ‗R' are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Mock exam 1 · Question 16

Question 133 | Q01502

The minimal set of super key is called

A. Foreign key

B. None of the choices is correct

C. Candidate key

D. Primary key

Hint

Test uniqueness and minimality separately when classifying keys.

Answer C: Candidate key

A candidate key is a minimal superkey. Minimal means no removable attribute while retaining uniqueness, not necessarily the fewest attributes among all keys.

Why each choice is right or wrong

A - Incorrect: Foreign key links relations, not the minimal-superkey definition. (PDF pp. 11.)

B - Incorrect: A matching definition is offered. (PDF pp. 11.)

C - Correct: Candidate key is the exact definition. (PDF pp. 11.)

D - Incorrect: Primary key is the selected candidate key rather than the category of every minimal superkey. (PDF pp. 11.)

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

> Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Question sources

Mock exam 1 · Question 18

Question 134 | Q01581

Assume every foreign-key column is NOT NULL and references a declared primary key. Which statement expresses referential integrity under these assumptions?

A. Every primary-key value must match a primary-key value in an associated table

B. Every primary-key value must match a foreign-key value in an associated table

C. Every foreign-key value must match a primary-key value in an associated table

D. Every foreign-key value must match a foreign-key value in an associated table

Hint

Match the validity requirement to the appropriate constraint definition.

Answer C: C. Every foreign-key value must match a primary-key value in an associated table

A foreign key is a reference from a child row to an eligible key in the parent. Because the question makes every reference non-NULL and targets primary keys, each foreign-key value must match a primary-key value. A parent row need not have children, and primary keys do not themselves have to reference another table. Merely matching a foreign-key column is not sufficient.

Why each choice is right or wrong

A - Incorrect: A primary key can exist without referencing another table.

B - Incorrect: A parent can have no child references.

C - Correct: Every non-NULL child reference must match its targeted parent primary key.

D - Incorrect: The target must be an eligible referenced key, not an arbitrary foreign-key column.

Primary note proof

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

> iv. Referential Integrity • A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key • Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Fundamental of Database Systems module | PDF page 49 | lines 2-3

- Sometimes a special NULL value may be used to create child entries without a parent or with a —dummy‖ parent

Question sources

Mock exam 3 · Question 13

Question 135 | Q01582

Which of the following is generally used for performing tasks like creating the structure of the relations, deleting relation?

A. DML(Data Manipulation Language)

B. Query

C. Relational Schema

D. DDL(Data Definition Language)

Hint

Classify each SQL command by the kind of change it performs.

Answer D: DDL(Data Definition Language)

Creating/deleting relation structure falls under DDL's structural command category. It is distinguished from data manipulation and from the schema description itself.

Why each choice is right or wrong

A - Incorrect: DML is the data-operation category. (PDF pp. 26.)

B - Incorrect: Query is not the schema-definition category. (PDF pp. 26.)

C - Incorrect: A relational schema is a description, not a command language. (PDF pp. 26.)

D - Correct: DDL is the named structural-language category. (PDF pp. 26.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Mock exam 3 · Question 14

Question 136 | Q01583

Which of the following is used in the application programs to request data from the database management system?

A. Data Manipulation language

B. Data Definition Language

C. Data Control Language

D. All choices are correct

Hint

Classify each SQL command by the kind of change it performs.

Answer A: Data Manipulation language

DML is the offered language category that includes retrieval requests such as SELECT. DDL handles definition, while authorization is separate from retrieval commands.

Why each choice is right or wrong

A - Correct: DML includes retrieval. (PDF pp. 26, 24.)

B - Incorrect: DDL is definition rather than ordinary data requests. (PDF pp. 26, 24.)

C - Incorrect: Access-control language is not the category that retrieves the requested data. (PDF pp. 26, 24.)

D - Incorrect: DDL and control are not all retrieval categories. (PDF pp. 26, 24.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 24 | lines 5-9

Relational algebra is the basic set of operations for the relational model. These operations enable a user to specify basic retrieval requests (or queries). The result of an operation is a new relation, which may have been formed from one or more input relations. The algebra operations thus produce new relations. Since each operation returns a relation, operations can be composed (Algebra is —closed‖.)

Question sources

Mock exam 3 · Question 15

Question 137 | Q01331B

Which single choice completely identifies the relational DBMS products in the list?

A. Oracle Database

B. Microsoft SQLServer

C. IBM Db2

D. All of the mentioned

Hint

Compare the data models by how they organize records and relationships.

Answer D: D. All of the mentioned

Oracle Database, Microsoft SQL Server and IBM Db2 are all relational DBMS products named in the note. Replacing the vendor-only IBM label with IBM Db2 makes every product entry well-defined. The question asks for the complete set, so All of the mentioned is the best answer. Each individual product is genuinely relational but identifies only part of the full set.

Why each choice is right or wrong

A - Incorrect: Oracle Database is relational, but this alone is an incomplete set.

B - Incorrect: SQL Server is relational, but this alone is an incomplete set.

C - Incorrect: IBM Db2 is relational, but this alone is an incomplete set.

D - Correct: The aggregate choice includes all three relational products.

Primary note proof

Study-note section FDB02

PostgreSQL, MySQL, Oracle Database, Microsoft SQL Server and IBM Db2 are relational DBMS products. MongoDB is a document DBMS. Java is a programming language. IBM alone names a vendor, not a database product. Distinguish a vendor such as IBM from a specific database product.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 14-16

Database Management System (DBMS) is then a tool for creating and managing databases. DBMS is a software that facilities the processes of defining, constructing, manipulating, and sharing databases. Example: MySQL server, MS ACESS, Microsoft SQL server.

Fundamental of Database Systems module | PDF page 13 | lines 18-27

3.6 The Relational Database Model The relational model represents the database as a collection of relations. A table is represented with columns and rows. Each table is an entity in the ER diagram. Each row is known as a tuple. Each table of the column has a name or attribute of the entity. 3.7 Relational Model Concepts in DBMS Attribute: Each column in a Table. Attributes are the properties which define a relation Tables - In the Relational model, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes. Tuple - It is a single row of a table, which contains a single record.

Question sources

Database Fundamentals practice questions · Question 6

Question 138 | Q01388B

In ER diagram Diamond/Ellipses represent

A. Relationship sets/Attributes

B. Entity sets/Attributes

C. Attributes/Relationships sets

D. Relationship sets/Entity sets

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer A: Diamonds represent relationship sets; ellipses represent attributes.

FDB06 maps rectangles to entity types, diamonds to relationships and ellipses to attributes in Chen notation. The question asks for the diamond/ellipse pair in that order. A preserves that order and meaning; the other choices either substitute entity sets or reverse the symbol meanings. A diagram-specific legend would override the generic convention.

Why each choice is right or wrong

A - Correct: The two shapes map to relationships and attributes respectively in Chen notation.

B - Incorrect: Entity types use rectangles, so a diamond does not represent the first item listed.

C - Incorrect: This reverses the requested meanings: diamonds indicate relationships, while ellipses indicate attributes.

D - Incorrect: The first meaning fits, but ellipses represent attributes rather than entity sets.

Primary note proof

Study-note section FDB06

> An entity is a distinguishable real or conceptual thing, such as one student or one course. An entity type describes common properties; its current collection of entities is an entity set. A relationship associates entities. In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes. Always inspect the diagram legend.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 10 | lines 3-13

> An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course). In ER diagrams, an entity is placed in a rectangle. Each entity has attributes, the particular properties that describe it. Entities that do not have key attributes of their own are called weak entity. The regular entity types that have a key attribute are called strong entity types. Weak entities are identified by being related to specific strong entities in combination with one of their attribute values. A weak entity type has a partial key, which is the attribute that can uniquely identify weak entities that are related to the same owner entity. In ER diagrams, both a weak entity type and its identifying relationship are distinguished by surrounding their boxes and diamonds with double lines. The partial key attribute is underlined with a dashed or dotted line.

Fundamental of Database Systems module | PDF page 11 | lines 13-15

> A relationship type represents the association between entity types. For example, _Enrolled in‘ is a relationship type that exists between entity Student and Course. In ER diagram, relationship is represented by a diamond and connecting the entities with lines.

Question sources

Database Fundamentals practice questions · Question 63

Question 139 | Q01388C

A relation is in 2NF if:

A. All the values of non-key attributes are dependent fully on the candidate key.

B. Any non-key attribute that are dependent on only part of the candidate key should be moved to another relation where the partial key is the actual full key.

C. It must be already in the 1NF.

D. All of the above.

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer D: All of the above.

Second normal form requires 1NF and full dependence of each non-prime attribute on candidate keys; partial dependencies must be moved to another relation. A states the criterion, B the repair, and C the prerequisite; D collects all three. A alone is not a complete definition without C.

Why each choice is right or wrong

A - Incorrect: Full dependency is a required 2NF property. (PDF pp. 17.)

B - Incorrect: Moving a partially dependent attribute to the relation where its determinant is the whole key is the stated repair. (PDF pp. 17.)

C - Incorrect: 1NF is expressly required. (PDF pp. 17.)

D - Correct: All three agree with the module's definition and repair rule. (PDF pp. 17.)

Primary note proof

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key. For Enrollment(SID,CID,SName,CTitle,Grade), assume key (SID,CID), SID→SName and CID→CTitle. Split into Student(SID,SName), Course(CID,CTitle) and Enrollment(SID,CID,Grade). Grade still depends on the whole pair. A relation whose every candidate key is single-attribute has no nonprime partial-key dependency. Higher forms inherit 2NF, so no partial dependencies establishes the minimum guaranteed stage only when 1NF is also assumed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 10-15

> Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Question sources

Database Fundamentals practice questions · Question 63

Question 140 | SM0017

Which one of the following is NOT a single level ordered index?

A. Secondary index

B. Primary index

C. Clustering index

D. Multilevel index

Hint

Compare index organization with its access benefits and maintenance costs.

Answer D: Multilevel index

In this single-level ordered-index classification, the listed types are primary, clustering and secondary indexes. A multilevel index is the exception to that listed single-level classification.

Why each choice is right or wrong

A - Incorrect: Secondary is listed as single level. (PDF pp. 21.)

B - Incorrect: Primary is listed as single level. (PDF pp. 21.)

C - Incorrect: Clustering is listed as single level. (PDF pp. 21.)

D - Correct: Multilevel is outside the specified single-level group. (PDF pp. 21.)

Primary note proof

Study-note section FDB25

> In ordered-file indexing terminology, a primary ordered index is on the file's ordering key and can have one entry per page, hence sparse. A clustering index is on a nonunique ordering field, often with an entry per distinct value. Secondary indexes provide other access paths and must represent all searchable values with record pointers or pointer lists. A dense index represents each search value; sparse indexes rely on ordered data between entries. Multilevel indexes index lower index levels. B/B+ trees support ordered and range search; hashing targets equality; bitmap indexes suit some analytic workloads. No one type is universally best.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 21 | lines 19-24

> Single Level Ordered Indexes Here it is assumed that a file already exists with some primary organization like unordered, ordered, or hashed organizations. For a file with a given record structure consisting of several fields, an index access structure is usually defined on a single field of a file, called an indexing field (or indexing attribute). Types of Single Level Ordered Indexes are Primary, Clustering and Secondary.

Question sources

MOE Computer Science Exit Exam · Question 5-1

National Computer Science Exit Exam · Question 32

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 42

Question 141 | SM0023

Among the following, which one best describes a Database Management System (DBMS)?

A. It is responsible for creating and maintaining a database

B. It can help users to manage computer's activity

C. It can assist process images

D. It can assist you to process document

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer A: It is responsible for creating and maintaining a database

A DBMS is software facilitating definition, construction, manipulation and sharing of databases. This directly matches creating and maintaining a database.

Why each choice is right or wrong

A - Correct: Matches the module's DBMS definition. (PDF pp. 5.)

B - Incorrect: Managing general computer activity describes a broader different role. (PDF pp. 5.)

C - Incorrect: Image processing is not the database-management role. (PDF pp. 5.)

D - Incorrect: Document processing is not the database-management definition. (PDF pp. 5.)

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 14-16

> Database Management System (DBMS) is then a tool for creating and managing databases. DBMS is a software that facilities the processes of defining, constructing, manipulating, and sharing databases. Example: MySQL server, MS ACESS, Microsoft SQL server.

Question sources

MOE Computer Science Exit Exam · Question 6-3

National Computer Science Exit Exam · Question 82

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 43

Question 142 | SM0034

Given the schemas: BOOK (Book_id, Title, Publisher_name), BOOK_AUTHOR (Book_id, Author_name), PUBLISHER (Name, Address, Phone). Which one of the following SQL is correct to delete the
publisher named 'XWZ' from the PUBLISHER table?

A. DELETE FROM BOOK_AUTHOR WHERE Address = 'XWZ'

B. DELETE FROM PUBLISHER WHERE Name = 'XWZ'

C. DELETE FROM PUBLISHER WHERE Address = 'XWZ'

D. DELETE FROM BOOK WHERE Name = 'XWZ'

Hint

Track which rows each data-changing command affects.

Answer B: DELETE FROM PUBLISHER WHERE Name = 'XWZ'

The module illustrates DELETE FROM HumanResources.JobCandidate WHERE JobCandidateID = 13 (PDF p.77). Substitute the requested relation PUBLISHER and its identifying Name predicate: DELETE FROM PUBLISHER WHERE Name = 'XWZ'. The stem's schemas distinguish Name from Address and omit both from BOOK_AUTHOR. This identifies the requested statement; successful execution may additionally depend on existing referencing rows and the deletion policy (PDF pp.48-49).

Why each choice is right or wrong

A - Incorrect: BOOK_AUTHOR is the wrong target and has no Address column in the supplied schema (compare the target/predicate form on PDF p.77).

B - Correct: PUBLISHER is the target and Name is the requested name filter, matching the module's DELETE FROM ... WHERE example (PDF p.77).

C - Incorrect: Filtering Address does not select by publisher name, despite a valid target table (PDF p.77's predicate pattern).

D - Incorrect: BOOK is the wrong target and has no Name column in the supplied schema (PDF p.77's target/predicate pattern).

Primary note proof

Study-note section FDB29

> INSERT INTO Hotel(HotelID,HotelName,SubCity) VALUES (1,'Hilton','Yeka'); pairs each named column with a compatible value. Counts and types must match; keys and other constraints still apply. Without a column list, values must follow the table's declared column order.

Study-note section FDB29

DELETE FROM Publisher WHERE Name = 'XWZ'; deletes matching publisher rows, subject to reference policies.

Study-note section FDB29

Both clear relation entries and delete rows describe that effect.

Related reading (context, not primary note proof)

Study-note section FDB16

Study-note section FDB15

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 77 | lines 14-18

Example 1: Commit Transaction BEGIN TRANSACTION; DELETE FROM HumanResources.JobCandidate WHERE JobCandidateID = 13; COMMIT;

Fundamental of Database Systems module | PDF page 48 | lines 13-17

iv. Referential Integrity ● A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key ● Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Fundamental of Database Systems module | PDF page 49 | lines 4-11

Deletion rules ● A row should not be deleted from the referenced table (parent) if there are matching rows in the referencing table (child) ● Three ways to handle this ◦ Restrict -- disallow the delete ◦ Nullify -- reset the foreign keys in the child to some NULL or dummy value ◦ Cascade -- Delete all rows in the child where there is a foreign key matching the key in the parent row being deleted

Question sources

MOE Computer Science Exit Exam · Question 10-3

National Computer Science Exit Exam · Question 9

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 44

Question 143 | SM0048

What does program data independence in the database approach entail?

A. The application may be affected when data changes

B. Data and applications are defined separately

C. Programs can create redundant data independently

D. Both program and data are defined together

Hint

Trace which schema levels a proposed change affects.

Answer B: Data and applications are defined separately

Program/data insulation separates application definitions from data definitions, allowing schema changes without corresponding higher-level changes under the relevant independence conditions.

Why each choice is right or wrong

A - Incorrect: Says changes necessarily affect applications, contrary to insulation. (PDF pp. 5, 6, 8.)

B - Correct: Separate definitions express program/data insulation. (PDF pp. 5, 6, 8.)

C - Incorrect: Creating redundant data independently is not the defined independence. (PDF pp. 5, 6, 8.)

D - Incorrect: Defining both together contradicts insulation. (PDF pp. 5, 6, 8.)

Primary note proof

Study-note section FDB04

> Physical data independence permits changing internal storage without changing the conceptual schema, for example adding an index. Logical data independence permits conceptual changes while preserving affected external views and application interfaces, for example splitting a table while maintaining a compatible view. It is harder because applications depend on logical meaning. Data abstraction hides irrelevant details; program-data insulation separates data definitions from application code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 22-25

> 1.3 Characteristics of the Database Approach The main characteristics of the database approach are Self-describing nature, Data abstraction and Insulation between programs and data, Support of multiple views of the data, sharing of data and multiuser transaction processing.

Fundamental of Database Systems module | PDF page 6 | lines 2-4

Data abstraction and Insulation between programs and data- Data Abstraction is a process of hiding unwanted or irrelevant details from the end user. It provides a different view and helps in achieving data independence which is used to enhance the security of data.

Fundamental of Database Systems module | PDF page 8 | lines 8-15

2.4 Data Independence The three-schema architecture can be used to further explain the concept of data independence, which can be defined as the capacity to change the schema at one level of a database system without having to change the schema at the next higher level. The two types of data independence: 1. Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs. 2. Physical data independence is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas do not need be changed as well.

Question sources

MOE Computer Science Exit Exam · Question 14-3

National Computer Science Exit Exam · Question 50

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 45

Question 144 | SM0071

Which statements are NOT correct descriptions of a fan trap or its general remedy in an ER diagram? Select all that apply. Treat B as a warning pattern, not a sufficient condition.

A. It occurs when there is no pathway between related entities

B. Highly likely when more than two one-to-many relationships fan out from an entity

C. When there is ambiguous pathway between some entity occurrences in related entity types

D. It can be resolved by readjusting the model so that no one-to-many relationships fan out from an entity

Hint

Trace the actual relationship paths and check whether the requested association follows.

Answer A, D: A. It occurs when there is no pathway between related entities; D. It can be resolved by readjusting the model so that no one-to-many relationships fan out from an entity

A fan trap concerns an ambiguous association inferred through branches from a common entity. A missing or broken path is instead a chasm-type issue. Also, legitimate one-to-many branches need not all be eliminated: the model should record the actual missing association. Multiple branches can warn of a fan trap, and ambiguity is its defining issue, so B and C are not false.

Why each choice is right or wrong

A - Correct: This is false about fan traps: a missing or broken path describes a chasm-type problem.

B - Incorrect: Several 1:N branches from a common entity can create the ambiguous association characteristic of fan traps.

C - Incorrect: Ambiguity about associations between entity occurrences is the defining fan-trap problem.

D - Correct: This is false as a general remedy: valid 1:N fans may remain after explicitly modeling the required association.

Primary note proof

Study-note section FDB14

> A fan trap occurs when two 1:N relationships fan from a common entity and a path suggests an association it cannot determine. Department→Employee and Department→Project do not reveal which employee works on which project; joining through Department pairs every employee with every project in that department. Record WorksOn explicitly when that is the fact needed.

Study-note section FDB14

> A missing path describes a chasm, not the defining ambiguity of a fan.

Study-note section FDB14

> Multiple 1:N relationships are legitimate; eliminating all fans is not a universal repair.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 27-31

> 3.3 ERD (Entity Relationship Diagram) ERD a graphical technique for representing a database schema. It is a modeling tool used to depict graphically a DB design before it is actually implemented. It shows the various entities being modeled and the important relationships among them. ERD has three basic components Entity, Relationship and Attribute.

Fundamental of Database Systems module | PDF page 11 | lines 24-31

> Cardinality The number of times an entity of an entity set participates in a relationship set is known as cardinality. Cardinality can be of different types: One to one (1-1) - When each entity in each entity set can take part only once in the relationship, the cardinality is one to one. One to many(1-M)- When entities in one entity set can take part only once in the relationship set and entities in other entity set can take part more than once in the relationship set, cardinality is many to one.

Question sources

MOE Computer Science Exit Exam · Question 20-3

National Computer Science Exit Exam · Question 24

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 46

Question 145 | SM0086

In which situation does using a DBMS become costly and not recommended?

A. In embedded systems having too small storage space

B. In situation where more than one user should access the database

C. To deny access to unauthorized users

D. To control redundancy

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer A: Severely storage-constrained embedded systems can make a DBMS too costly.

FDB01 says storage, administration and complexity may outweigh DBMS benefits in a tiny fixed application, while also recognizing lightweight embedded DBMSs. A is therefore the listed situation where resource constraints can make a DBMS inappropriate. The other scenarios are central reasons for using a DBMS: sharing, authorization and redundancy control. Embedded use alone does not forbid a DBMS.

Why each choice is right or wrong

A - Correct: Severely limited storage can make DBMS overhead unjustified, though lightweight embedded DBMSs may still fit.

B - Incorrect: Controlled multiuser access is a benefit supplied by database management rather than a reason to avoid it.

C - Incorrect: Authorization helps deny unauthorized access and is a reason to use DBMS protection.

D - Incorrect: Controlling redundancy is a benefit of shared database management rather than a contraindication.

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB01

Complexity, administration, storage and licensing can outweigh benefits for a tiny fixed application; lightweight embedded DBMSs also exist.

Study-note section FDB01

It controls redundancy rather than guaranteeing zero duplication.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 14-25

Database Management System (DBMS) is then a tool for creating and managing databases. DBMS is a software that facilities the processes of defining, constructing, manipulating, and sharing databases. Example: MySQL server, MS ACESS, Microsoft SQL server. 1.2 Database System versus File System The traditional file processing system is file-directory structure supported by a conventional operating system. A file system organization of data lucks a number of major features of a database system, such as Data redundancy and inconsistency, Difficulty in accessing data, Data isolation, Integrity problems, Atomicity problems, Concurrent access anomalies. 1.3 Characteristics of the Database Approach The main characteristics of the database approach are Self-describing nature, Data abstraction and Insulation between programs and data, Support of multiple views of the data, sharing of data and multiuser transaction processing.

Fundamental of Database Systems module | PDF page 19 | lines 4-10

5.1 Introduction To translate the logical description of data into technical specifications for storing and retrieving data, a physical database design is needed. The goal is to create a design for storing and retrieving data that will provide adequate performance and insure database integrity, security, and availability. Some decisions need to be made on the physical database design. They are Attribute data types, data/file storages, File organizations and Indexes.

Question sources

MOE Computer Science Exit Exam · Question 24-3

National Computer Science Exit Exam · Question 72

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 47

Question 146 | SM0208

Indexes in database tables speed up the _____ operations.

A. Read operations

B. Delete operations

C. Update operations

D. Write operations

Hint

Compare index organization with its access benefits and maintenance costs.

Answer A: Read operations

Indexes are auxiliary access structures explicitly used to speed record retrieval for search conditions. This is a read-performance purpose; it does not claim that maintaining indexes makes every change operation faster.

Why each choice is right or wrong

A - Correct: Read retrieval is the stated purpose. (PDF pp. 21.)

B - Incorrect: Deletion may require maintenance; it is not the stated general benefit. (PDF pp. 21.)

C - Incorrect: Updating data is not the stated general purpose. (PDF pp. 21.)

D - Incorrect: Writing is not the stated retrieval purpose. (PDF pp. 21.)

Primary note proof

Study-note section FDB25

> An index stores search keys with pointers or row identifiers. It often speeds selective retrieval but consumes space and must be maintained on writes. A full scan can win for a tiny table or a query returning most rows. Indexes do not themselves establish referential integrity.

Study-note section FDB25

> Secondary indexes provide other access paths and must represent all searchable values with record pointers or pointer lists.

Study-note section FDB25

> A dense index represents each search value; sparse indexes rely on ordered data between entries.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 21 | lines 5-9

> 5.5 Index Indexes are additional auxiliary access structures which are used to speed up the retrieval of records in response to certain search conditions. Index structures are additional files, which provide alternative ways to access the records without affecting the physical placement of records on disk.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 3-1

Computer Science Exit Exam practice collection · Question 56

Question 147 | SM0226

An e-commerce database already has a suitable normalized schema. Customer-order queries repeatedly select a small fraction of rows by a search key. Which technique directly adds a maintained search path for those lookups?

A. Normalization

B. Denormalization

C. Partitioning

D. Indexing

Hint

Compare design techniques by the specific problem each solves.

Answer D: D. Indexing

Indexing stores search keys and pointers that support selective retrieval without scanning every row. That directly meets the requested search-path improvement. Normalization addresses dependency anomalies, denormalization stores redundant or precomputed facts, and partitioning divides a dataset. Those techniques can be useful elsewhere, but the specified task is to add an access path to the existing normalized schema.

Why each choice is right or wrong

A - Incorrect: Normalization organizes dependencies; it does not add the requested search path.

B - Incorrect: Denormalization repeats or precomputes facts rather than directly adding an index path.

C - Incorrect: Partitioning divides the dataset rather than adding the requested maintained search-key structure.

D - Correct: Indexing directly supplies the selective lookup search path.

Primary note proof

Study-note section FDB38

> Normalization reduces dependency-driven anomalies. Indexing supplies search paths. Denormalization deliberately repeats or precomputes facts to reduce selected read costs, increasing maintenance work. Partitioning divides rows or columns; replication creates copies to improve locality or availability while complicating updates. These techniques can coexist. An e-commerce or gaming label alone does not select one.

Study-note section FDB25

> An index stores search keys with pointers or row identifiers.

Study-note section FDB25

> It often speeds selective retrieval but consumes space and must be maintained on writes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 17-27

> 4.4 Normalization Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. It is a technique of organizing the data in the database. It is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables. Normal forms Redundancy in a relation may causes insertion, deletion, and update anomalies. Normalization is database design technique that minimizes redundancy from a relation or a set of relations. Normalization rules divides larger tables into smaller tables and links them using relationship.

Fundamental of Database Systems module | PDF page 21 | lines 5-9

> 5.5 Index Indexes are additional auxiliary access structures which are used to speed up the retrieval of records in response to certain search conditions. Index structures are additional files, which provide alternative ways to access the records without affecting the physical placement of records on disk.

Fundamental of Database Systems module | PDF page 117 | lines 18-29

> I. Data Replication • The system maintain several identical copies of the relation & store each copy at a different site • In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead II. Data Fragmentation - Split a relation into logically related and correct parts. - The main reasons for fragmenting a relation are • Efficiency- data that is not needed by the local applications is not stored • Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency

Question sources

Computer Science Exit Exam · Sene 2016 · Question 9-1

Computer Science Exit Exam practice collection · Question 68

Question 148 | SM0251

Which SQL command is used to modify existing data in a table?

A. ALTER

B. UPDATE

C. MODIFY

D. CHANGE

Hint

Track which rows each data-changing command affects.

Answer B: UPDATE

UPDATE changes existing row values using a target table, SET assignments and an optional WHERE predicate. ALTER instead changes an object's definition. The bare words MODIFY and CHANGE do not form the UPDATE statement required for changing stored row values in the stated SQL forms. They should not be treated as synonyms for the command merely because they describe the task in everyday language. B names the actual row-update command.

Why each choice is right or wrong

A - Incorrect: Incorrect: ALTER changes schema definitions rather than stored row values.

B - Correct: Correct: UPDATE changes existing rows using SET and, when needed, WHERE.

C - Incorrect: Incorrect: MODIFY is not the UPDATE command required for the stated row-value operation.

D - Incorrect: Incorrect: CHANGE is not the UPDATE command required for the stated row-value operation.

Primary note proof

Study-note section FDB29

> UPDATE Book SET Price = 600 WHERE ID = 'B001'; changes matching rows. UPDATE without WHERE targets every row.

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 7-11

> 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 54 | lines 16-18

> • <privileges> is a list of SELECT <columns>, INSERT <columns>, DELETE, and UPDATE <columns>, or simply ALL

Fundamental of Database Systems module | PDF page 55 | lines 2-5

> UPDATE(Salary) ON Employee TO Finance The user ‗Finance' can view the entire Employee table, and can change Salary values, but cannot change other values or pass on their privilege

Question sources

Computer Science Exit Exam · Sene 2016 · Question 16-4

Computer Science Exit Exam practice collection · Question 54

Question 149 | SM0252

Which is the first normal form in the usual normalization sequence that requires atomic values and no repeating groups?

A. Second Normal Form (2NF)

B. Third Normal Form (3NF)

C. Boyce-Codd Normal Form (BCNF)

D. First Normal Form (1NF)

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer D: First Normal Form (1NF)

First normal form is the first stage requiring atomic values and no repeating groups in cells. It does not by itself remove partial or transitive functional dependencies. Second normal form adds the full-dependency condition, third normal form adds the determinant/prime-attribute rule, and BCNF requires every nontrivial determinant to be a superkey. These stronger forms inherit atomicity, but the question asks for the first stage defining it.

Why each choice is right or wrong

A - Incorrect: Incorrect: 2NF inherits atomicity and additionally excludes nonprime partial dependencies; it is a later stage.

B - Incorrect: Incorrect: 3NF inherits the earlier atomicity condition and adds a stronger dependency test.

C - Incorrect: Incorrect: BCNF adds the superkey-determinant rule and is not the first stage requiring atomicity.

D - Correct: Correct: 1NF is the first stage defining atomic values and no repeating groups.

Primary note proof

Study-note section FDB20

> 1NF requires atomic values in the chosen relational domains and no repeating groups in cells.

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Study-note section FDB21

> For every nontrivial FD X→A, 3NF permits it when X is a superkey or A is prime. BCNF requires X to be a superkey. The mnemonic whole key and nothing but the key describes the common single-key case; it cannot replace the full tests when keys overlap. Remove dependencies of nonprime facts on nonkey determinants.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 2-15

> First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF). Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

> Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 17-1

Question 150 | SM0254

What is the purpose of normalization in database design?

A. To increase data redundancy

B. To decrease data integrity

C. To eliminate data duplication

D. To simplify data retrieval

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer C: To eliminate data duplication

Normalization systematically decomposes relations to eliminate or reduce duplication and insertion/update/deletion anomalies. Eliminate is understood as the design goal, not a claim that every possible repeated value or necessary key reference disappears.

Why each choice is right or wrong

A - Incorrect: Increasing redundancy reverses the goal. (PDF pp. 16.)

B - Incorrect: Decreasing integrity reverses the anomaly-prevention goal. (PDF pp. 16.)

C - Correct: Eliminating duplicated data is directly stated. (PDF pp. 16.)

D - Incorrect: Simplifying retrieval is not the stated normalization purpose and does not follow automatically from decomposition. (PDF pp. 16.)

Primary note proof

Study-note section FDB19

> Suppose Enrollment repeats StudentName for every course. An update anomaly requires changing every copy of the student's name; missed copies disagree. An insertion anomaly prevents recording a new student until some enrollment exists. A deletion anomaly loses the only record of a student's details when their last enrollment is deleted. Revoke anomaly is not a normalization anomaly.

Study-note section FDB19

> Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design.

Study-note section FDB19

> Reduced redundancy can improve integrity and storage use, but extra keys and joins mean neither storage savings nor faster retrieval is guaranteed.

Related reading (context, not primary note proof)

Study-note section FDB38

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 17-29

> 4.4 Normalization Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. It is a technique of organizing the data in the database. It is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables. Normal forms Redundancy in a relation may causes insertion, deletion, and update anomalies. Normalization is database design technique that minimizes redundancy from a relation or a set of relations. Normalization rules divides larger tables into smaller tables and links them using relationship. Normal Forms are maintenance rules that are used to eliminate or reduce the redundancy in database relations. These normal forms are 1NF (First Normal Form), 2NF (Second Normal Form)

Variant 1 | Alternative version 1

What is the purpose of normalization in database design?

A. To reduce redundancy and improve data integrity

B. To speed up query performance

C. To create complex relationships between tables

D. To encrypt sensitive data

Answer A: To reduce redundancy and improve data integrity

Normalization systematically decomposes relations to eliminate or reduce duplication and insertion/update/deletion anomalies. Eliminate is understood as the design goal, not a claim that every possible repeated value or necessary key reference disappears.

Why each choice is right or wrong

A - Correct: Reducing redundancy and preventing anomalies is the normalization goal (PDF p.16).

B - Incorrect: Query performance is not the stated normalization purpose; decomposition can require additional joins.

C - Incorrect: Relationships support decomposition but creating complexity is not its goal.

D - Incorrect: Encryption is a security mechanism, not normalization.

Primary note proof

Study-note section FDB19

> Suppose Enrollment repeats StudentName for every course. An update anomaly requires changing every copy of the student's name; missed copies disagree. An insertion anomaly prevents recording a new student until some enrollment exists. A deletion anomaly loses the only record of a student's details when their last enrollment is deleted. Revoke anomaly is not a normalization anomaly.

Study-note section FDB19

> Reduced redundancy can improve integrity and storage use, but extra keys and joins mean neither storage savings nor faster retrieval is guaranteed.

Study-note section FDB19

> Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design.

Related reading (context, not primary note proof)

Study-note section FDB38

Question sources

Computer Science Exit Exam · Sene 2016 · Question 17-3

Computer Science MCQ practice collection · Question 131

Computer Science MCQ practice collection · Question 152

Computer Science Exit Exam practice collection · Question 53

Question 151 | SM0262

Which of the following IS NOT a type of database management system (DBMS)?

A. PostgreSQL

B. MongoDB

C. Java

D. MySQL

Hint

Compare the data models by how they organize records and relationships.

Answer C: Java is a programming language, not a DBMS.

The question uses type loosely to list named systems. FDB02 identifies PostgreSQL and MySQL as relational DBMS products and MongoDB as a document DBMS. Java is explicitly a programming language. Its ability to connect to databases does not make it a database management system, so C alone fits the negative question.

Why each choice is right or wrong

A - Incorrect: PostgreSQL is a relational DBMS product, so it does not satisfy the negative question.

B - Incorrect: MongoDB is a document DBMS, even though it is not a relational system.

C - Correct: Java is a programming language rather than a database management system.

D - Incorrect: MySQL is a relational DBMS product and therefore is not the requested exception.

Primary note proof

Study-note section FDB02

> PostgreSQL, MySQL, Oracle Database, Microsoft SQL Server and IBM Db2 are relational DBMS products. MongoDB is a document DBMS. Java is a programming language. IBM alone names a vendor, not a database product. Distinguish a vendor such as IBM from a specific database product.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 14-16

> Database Management System (DBMS) is then a tool for creating and managing databases. DBMS is a software that facilities the processes of defining, constructing, manipulating, and sharing databases. Example: MySQL server, MS ACESS, Microsoft SQL server.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 20-2

Computer Science Exit Exam practice collection · Question 57

Question 152 | SM0273

The E-R data model is based on a perception of the real world that consists of a set of basic objects called _____.

A. Objects

B. Classes

C. Entities

D. Attributes

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer C: Entities

The ER model uses entities, attributes and relationships; the actual physical/conceptual objects are entities. Objects is a generic word, while entities is the model-specific category requested.

Why each choice is right or wrong

A - Incorrect: Objects is the generic description, not the requested ER term. (PDF pp. 9, 10.)

B - Incorrect: Classes are not the basic-object term in this ER definition. (PDF pp. 9, 10.)

C - Correct: Entities is the named category for those real-world objects. (PDF pp. 9, 10.)

D - Incorrect: Attributes are properties of entities rather than the objects themselves. (PDF pp. 9, 10.)

Primary note proof

Study-note section FDB06

> An entity is a distinguishable real or conceptual thing, such as one student or one course. An entity type describes common properties; its current collection of entities is an entity set. A relationship associates entities. In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes. Always inspect the diagram legend.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 3-6

> 3.1 Database Modeling High-level conceptual data models provide concepts for presenting data in ways that are close to the way people perceive data. A typical example is the entity relationship model, which uses main concepts like entities, attributes and relationships.

Fundamental of Database Systems module | PDF page 10 | lines 3-8

> An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course). In ER diagrams, an entity is placed in a rectangle. Each entity has

attributes, the particular properties that describe it. Entities that do not have key attributes of their own are called weak entity. The regular entity types that have a key attribute are called strong entity types. Weak entities are identified by being related to specific strong entities

Question sources

Computer Science Exit Exam · Sene 2016 · Question 24-1

Computer Science Exit Exam practice collection · Question 52

Question 153 | SM0275

What is the primary key in a relational database?

A. A unique identifier for each row in a table

B. A foreign key from another table

C. A key used for encryption

D. An index on a non-unique column

Hint

Test uniqueness and minimality separately when classifying keys.

Answer A: A unique identifier for each row in a table

A primary key is the selected candidate key used to identify tuples uniquely; its values must be unique and non-null. The key may consist of more than one attribute.

Why each choice is right or wrong

A - Correct: Uniquely identifies each row, consistent with the key definition. (PDF pp. 11, 14.)

B - Incorrect: A foreign key serves a reference role and need not be this table's primary key. (PDF pp. 11, 14.)

C - Incorrect: Encryption keys have a different security role. (PDF pp. 11, 14.)

D - Incorrect: A non-unique index does not meet key uniqueness. (PDF pp. 11, 14.)

Primary note proof

Study-note section FDB07

A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

A table can have several candidates and foreign keys but at most one declared primary-key constraint, possibly containing several columns.

Study-note section FDB07

> If ID is unique, both {ID} and {ID,City} are superkeys, but only {ID} is minimal.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

> Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Fundamental of Database Systems module | PDF page 14 | lines 17-20

> Key Constraints- These are called uniqueness constraints since it ensures that every tuple in the relation should be unique. An attribute that can uniquely identify a tuple in a relation is called the key or primary key of the table. The value of the attribute for different tuples in the relation has to be unique. Primary keys must be unique and cannot be null.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 24-3

Computer Science Exit Exam practice collection · Question 51

Question 154 | SM0280

What is the role of a database administrator (DBA)?

A. Designing user interfaces for database applications

B. Developing front-end applications

C. Writing complex SQL queries

D. Managing and maintaining databases

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer D: Managing and maintaining databases

The DBA administers the database and DBMS, authorizes access, coordinates/monitors use and acquires resources. Those responsibilities fit management and maintenance.

Why each choice is right or wrong

A - Incorrect: User-interface design is not the defining DBA responsibility described. (PDF pp. 6.)

B - Incorrect: Front-end development is not the defining responsibility. (PDF pp. 6.)

C - Incorrect: Writing a difficult query may occur, but is not the administration role defined. (PDF pp. 6.)

D - Correct: Management and maintenance match the listed administration duties. (PDF pp. 6.)

Primary note proof

Study-note section FDB01

> The DBA manages access authorization, backup/recovery, availability, monitoring, tuning and routine maintenance, coordinating database use; designers define the schema and application developers build user-facing programs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 6 | lines 10-16

> 1.4 Actors on the scene The people whose jobs involve the day-to-day use of a large database are called actors on the scene. In a database environment, the primary resource is the database itself, and the secondary resource is the DBMS and related software. Administering these resources is the responsibility of the database administrator (DBA). The DBA is responsible for authorizing access to the database, coordinating and monitoring its use, and acquiring software and hardware resources as needed.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 26-2

Computer Science Exit Exam practice collection · Question 55

Question 155 | SM0304

Which of the following IS NOT a type of database model?

A. Linear

B. Object-Oriented

C. Hierarchical

D. Relational

Hint

Compare the data models by how they organize records and relationships.

Answer A: Linear

The model taxonomy names object-oriented, hierarchical/network, and relational concepts. Linear is not among the stated database-model categories; this is a taxonomy question within that stated list, not a universal claim about every possible model name.

Why each choice is right or wrong

A - Correct: Linear is outside the supplied taxonomy. (PDF pp. 7, 13.)

B - Incorrect: Object-Oriented is explicitly a conceptual-model example. (PDF pp. 7, 13.)

C - Incorrect: Hierarchical is named in the model discussion. (PDF pp. 7, 13.)

D - Incorrect: Relational is explicitly defined as a database model. (PDF pp. 7, 13.)

Primary note proof

Study-note section FDB02

A data model supplies concepts for structure, relationships, constraints and operations. Conceptual ER models describe entities and relationships. Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents. Object models represent objects with identity, state and behavior. A physical model specifies record layouts and access paths. These are different representation choices, not stages of one query.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 8-19

Types of Data Models 1. High-level or conceptual data models provide concepts that are close to the way many users perceive data. Entity - Relationship (E/R) model and Object-Oriented Model are examples of conceptual/high level data model. 2. Low-level or physical data models provide concepts that describe the details of how data is stored on the computer storage media. Concepts provided by physical data models are generally meant for computer specialists, not for end users. 3. representational (or implementation) data model provides concepts that may be easily understood by end users but that are not too far from the way data is organized in computer storage. Representational data models hide many details of data storage on disk but can be implemented on a computer system directly. The widely used relational data models are the network and hierarchical.

Fundamental of Database Systems module | PDF page 13 | lines 18-20

3.6 The Relational Database Model The relational model represents the database as a collection of relations. A table is represented with columns and rows. Each table is an entity in the ER diagram. Each row is known as a tuple.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 34-1

Computer Science Exit Exam practice collection · Question 58

Question 156 | SS133

A relation has atomic values with no repeating groups, and no nonprime attribute partially depends on any candidate key. What is the strongest normal form guaranteed by these facts alone?

A. 4NF

B. 2NF

C. 1NF

D. 3NF

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer B: B. 2NF

Atomic values without repeating groups establish 1NF. Adding the absence of nonprime partial dependencies establishes 2NF. Higher normal forms have additional requirements involving nonkey determinants or multivalued dependencies, and none of those requirements follows from the stated facts. Thus 2NF is the strongest guaranteed level, even though the relation could also happen to satisfy higher forms.

Why each choice is right or wrong

A - Incorrect: No restriction on nontrivial multivalued dependencies has been established.

B - Correct: 1NF plus the absence of nonprime partial dependencies guarantees 2NF.

C - Incorrect: 1NF holds, but 2NF is the stronger guarantee.

D - Incorrect: 3NF requires additional determinant/prime-attribute conditions.

Primary note proof

Study-note section FDB20

> 1NF requires atomic values in the chosen relational domains and no repeating groups in cells. It does not mean merely declaring a primary key. A mapped ER design that correctly splits composite and multivalued structures gives ordinary 1NF tables; ER drawing alone guarantees no higher form.

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Study-note section FDB20

> Higher forms inherit 2NF, so no partial dependencies establishes the minimum guaranteed stage only when 1NF is also assumed.

Related reading (context, not primary note proof)

Study-note section FDB21

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 2-15

> First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF). Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

> Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Computer Science model exam collection · Question 40

Question 157 | SS142

Which one of the following is not among the characterizing features of database approach?

A. Sharing of data

B. @-tTM Application-data dependency

C. j Oc. Self-describing

D. Data abstraction

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer B: @-tTM Application-data dependency

Database systems support sharing, self-description and abstraction/insulation. Application-data dependency is the opposite of insulation and independence.

Why each choice is right or wrong

A - Incorrect: Sharing is expressly a characteristic. (PDF pp. 5, 6.)

B - Correct: Application-data dependency contradicts the stated insulation characteristic. (PDF pp. 5, 6.)

C - Incorrect: Self-description is expressly a characteristic. (PDF pp. 5, 6.)

D - Incorrect: Abstraction is expressly a characteristic. (PDF pp. 5, 6.)

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB04

> Data abstraction hides irrelevant details; program-data insulation separates data definitions from application code.

Study-note section FDB04

> Logical data independence permits conceptual changes while preserving affected external views and application interfaces, for example splitting a table while maintaining a compatible view.

Related reading (context, not primary note proof)

Study-note section FDB03

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 22-29

> 1.3 Characteristics of the Database Approach The main characteristics of the database approach are Self-describing nature, Data abstraction and Insulation between programs and data, Support of multiple views of the data, sharing of data and multiuser transaction processing.
> Self-describing nature - A fundamental characteristic of the database approach is that the database system contains not only the database itself (data) but also a complete definition or description of the database structure and constraints. This information is called meta-data, and it describes the structure of the database.

Fundamental of Database Systems module | PDF page 6 | lines 2-9

> Data abstraction and Insulation between programs and data- Data Abstraction is a process of hiding unwanted or irrelevant details from the end user. It provides a different view and helps in achieving data independence which is used to enhance the security of data. Support of multiple views of the data - A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored. Sharing of data and multiuser transaction processing - A multiuser DBMS must allow multiple users to access the database at the same time. This is essential if data for multiple applications is to be integrated and maintained in a single database.

Question sources

Computer Science model exam collection · Question 49

Question 158 | SS146

"Database is a logically coherent collection of data with some inherent meaning" describes that

A. the entity should be physical object

B. number of attributes should be limited

C. there should be one primary key attribute

D. all attributes of the entity should be related

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer D: all attributes of the entity should be related

A database is a collection of related data with implicit meaning. D is the closest reading of coherence in the question: attributes should describe related subject matter, not arbitrary unrelated data. It does not mean every attribute functionally determines every other.

Why each choice is right or wrong

A - Incorrect: Entities may have conceptual existence too. (PDF pp. 5, 10, 11.)

B - Incorrect: No bound on the number of attributes is implied. (PDF pp. 5, 10, 11.)

C - Incorrect: A primary key may have multiple attributes; coherence does not require a single attribute. (PDF pp. 5, 10, 11.)

D - Correct: Related attributes are the offered interpretation consistent with meaningful related data. (PDF pp. 5, 10, 11.)

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB07

> A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB06

> An entity type describes common properties; its current collection of entities is an entity set.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 5-9

> Data is known facts that can be recorded and that have implicit meaning. Database is a collection of related data. For example, consider the names, telephone numbers, and addresses of the people you know. Nowadays, this data is typically stored in mobile phones, which have their own simple database software. This collection of related data with an implicit meaning is a database. A database can be defined as a self-describing collection of integrated records.

Fundamental of Database Systems module | PDF page 10 | lines 3-5

> An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course). In ER diagrams, an entity is placed in a rectangle.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

> Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Question sources

Computer Science model exam collection · Question 53

Question 159 | SS157

For HOTEL(Hotel_id INTEGER PRIMARY KEY, Hotel_Name STRING, Sub_city STRING), which statement correctly inserts one row? Assume Hotel_id = 1 is unused and no other constraints prevent the insert.

A. INSERT INTO HOTEL(Hotel_id,Hotel_Name,Sub_city) Values('Hilton', Yeka')

B. INSERT INTO HOTEL(Hotel_id,Hotel_Name,Sub_city) Values (1,'Hilton','Yeka')

C. INSERT INTO HOTEL(1,'Hilton','Yeka')

D. INSERT Values INTO HOTEL(Hotel_id,Hotel_Name,Sub_city) Values(1,'Hilton','Yeka')

Hint

Track which rows each data-changing command affects.

Answer B: INSERT INTO HOTEL(Hotel_id,Hotel_Name,Sub_city) Values (1,'Hilton','Yeka')

An INSERT must match each named column with a compatible value and follow INSERT INTO ... VALUES syntax. B supplies the integer identifier and the two text values in the named order; the stated unused-key assumption removes a possible primary-key conflict. A supplies too few values and has damaged string quoting. C puts values where column names belong and omits VALUES, while D places an extra Values token before INTO. These are incorrect statements, not unresolved alternatives.

Why each choice is right or wrong

A - Incorrect: Incorrect: three named columns require three compatible values; this text supplies only two and also has malformed quoting.

B - Correct: Correct: it supplies an unused integer key and two strings in the named column order with the required INSERT syntax.

C - Incorrect: Incorrect: literal values occupy the column-list position and the VALUES clause is absent.

D - Incorrect: Incorrect: INSERT Values INTO has an extra token before INTO and breaks the required INSERT structure.

Primary note proof

Study-note section FDB29

> INSERT INTO Hotel(HotelID,HotelName,SubCity) VALUES (1,'Hilton','Yeka'); pairs each named column with a compatible value. Counts and types must match; keys and other constraints still apply. Without a column list, values must follow the table's declared column order.

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Related reading (context, not primary note proof)

Study-note section FDB07

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 77 | lines 19-24

> Example 2: Rollback Transaction CREATE TABLE ValueTable (id int); BEGIN TRANSACTION; INSERT INTO ValueTable VALUES(1); INSERT INTO ValueTable VALUES(2); ROLLBACK;

Fundamental of Database Systems module | PDF page 14 | lines 14-20

> Domain Constraints - Every domain (cell in the table or attribute value) must contain atomic values (smallest indivisible units). We perform data type check here, which means we need to assign a data type to a column and limit the values that it can contain. Key Constraints- These are called uniqueness constraints since it ensures that every tuple in the relation should be unique. An attribute that can uniquely identify a tuple in a relation is called the key or primary key of the table. The value of the attribute for different tuples in the relation has to be unique. Primary keys must be unique and cannot be null.

Fundamental of Database Systems module | PDF page 19 | lines 11-19

5.2 Choosing Attribute Data Types CHAR - fixed-length character VARCHAR() - variable-length characters DATE - actual date BLOB - binary large object (good for graphics images, audio, sound clips, etc.) NUMERIC - positive/negative numbers Exact • SMALLINT → 16 bit • INT → 32 bit

Question sources

Computer Science model exam collection · Question 64

Question 160 | SS172

DORM(DormID, FloorNumber) has primary key DormID. STUDENT(IdNo, Name, Department, DormID) has primary key IdNo. Which table-constraint fragment correctly declares STUDENT.DormID as a reference to DORM? Capacity six is enforced separately; the surrounding CREATE TABLE syntax is supplied.

A. FOREIGN KEY (DormID) REFERENCED BY DORM(DormID)

B. DORM table is a referencing relation

C. FOREIGN KEY (DormID) REFERENCES DORM(DormID)

D. STUDENT table is a referenced relation

Hint

Trace keys and cardinalities through the integrated schema.

Answer C: C. FOREIGN KEY (DormID) REFERENCES DORM(DormID)

The student row stores the dorm identifier, making STUDENT the referencing child and DORM the referenced parent. The required clause names the child column and uses REFERENCES to identify Dorm's key. C has this syntax after the naming repair. The foreign key validates a dorm reference; it does not by itself enforce the six-student capacity rule, which the question handles separately.

Why each choice is right or wrong

A - Incorrect: REFERENCED BY is not the required child-to-parent clause.

B - Incorrect: DORM is the referenced parent, not the referencing child.

C - Correct: REFERENCES DORM(DormID) correctly targets the parent key.

D - Incorrect: STUDENT is the referencing child.

Primary note proof

Study-note section FDB39

Use Department(DeptID,Name), Student(SID,Name,DeptID,DormID), Course(CID,Title), Enrollment(SID,CID,Grade) and Dorm(DormID,FloorNumber). Declare all entity IDs primary keys; Student.DeptID and Student.DormID reference their parents; Enrollment has composite key (SID,CID) and both references. Make DormID nullable until assignment. Separate StudentPhone(SID,Phone) for multiple contacts. Normalize repeated department/course facts into their owning tables. A simple foreign key does not enforce the six-student capacity; use a

concurrency-safe trigger, allocation transaction or explicit slot design with six unique slots per dorm.

Study-note section FDB39

STUDENT is the referencing child; DORM is referenced.

Study-note section FDB39

Consistent names and an added Student.DormID column are required before execution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 12 | lines 18-21

4. For each binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type. Include the primary key of the relation T as foreign key in S that represents the other entity type participating in R. Include any attributes of the 1:N relation type as attributes of S.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

iv. Referential Integrity • A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key • Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Fundamental of Database Systems module | PDF page 49 | lines 23-28

Referential Integrity • Ensures that dependent relationships in the data are maintained. In Oracle, for example: • CREATE TABLE table-name (attr1 attr-type PRIMARY KEY, attr2 attr-type NOT NULL, ..., attrM attr-type REFERENCES owner.tablename(attrname) ON DELETE CASCADE, ...

Question sources

Computer Science model exam collection · Question 79

Question 161 | SS189

Which one of the following is an appropriate sequence of database design processes?

A. Logical database design, Enterprise data modeling, Physical database design, Database implementation

B. Enterprise data modeling, Logical database design, Database implementation, Physical database design

C. Logical database design, Enterprise data modeling, Physical database design, Database implementation

D. Enterprise data modeling, Logical database design, Physical database design, Database implementation

Hint

Identify the design phase from its inputs and intended deliverable.

Answer D: Enterprise modeling → logical design → physical design → implementation.

FDB05 starts with business requirements and conceptual ER modeling, then maps to relations and constraints in logical design, chooses storage and indexes in physical design, and creates/loads actual objects in implementation. D preserves that order. The other choices either begin logical design before enterprise modeling or place implementation before the storage decisions it must implement.

Why each choice is right or wrong

A - Incorrect: Logical design should use the conceptual enterprise model rather than precede its creation.

B - Incorrect: Physical storage decisions precede implementation, not follow it in the stated design sequence.

C - Incorrect: This duplicates A and again places logical design before enterprise modeling.

D - Correct: It follows conceptual enterprise modeling, relational mapping, storage design and actual implementation.

Primary note proof

Study-note section FDB05

> First collect business facts, required transactions, reports, volumes, privacy rules and quality targets. Conceptual design turns them into an ER/EER model. Logical design maps the conceptual model to relations, keys and constraints and checks normalization. Physical design chooses file organization, indexes and storage. Implementation creates and loads the actual objects; testing checks requirements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 7-26

> 3.2 Phases of database design The phases of database development life cycle (DDLC) in the Database Management System (DBMS) are explained below. Requirement Analysis The most important step in implementing a database system is to find out what is needed i.e what type of a database is required for the business organization, daily volume of data, how much data needs to be stored in the master files etc. In order to collect all this information, a database analyst spends a lot of time within the business organization talking to people, end users and getting acquainted with the day-to-day process. Conceptual Database Design In this phase the database designers will make a decision on the database model that perfectly suits the organization‘s requirement. The database designers will study the documents prepared by the analysis in the requirement analysis stage and then start development of a system model that fulfils the needs. Logical database design (Data model mapping) The conceptual design is translated into internal model which includes mapping of all objects i.e design of tables, indexes, views, transaction, access privileges etc. Physical Design Physical design or Database implementation needs the

formation of special storage related constructs. These constructs consist of storage groups, table spaces, data files, tables etc.

Question sources

Computer Science model exam collection · Question 96

Question 162 | AM0132

Which of the following is a primary key in a relational database?

A. A unique identifier for a row in a table

B. A foreign key from another table

C. A non-unique attribute in a table

D. A combination of attributes in a table

Hint

Test uniqueness and minimality separately when classifying keys.

Answer A: A unique identifier for a row in a table

A primary key is selected from candidate keys to uniquely identify a row and must be unique/non-null. Multiple attributes alone do not make a set a primary key unless those requirements and selection hold.

Why each choice is right or wrong

A - Correct: Gives the essential identifying property. (PDF pp. 11, 14.)

B - Incorrect: A foreign key is a reference and is not necessarily the selected primary identifier. (PDF pp. 11, 14.)

C - Incorrect: Non-unique attributes cannot uniquely identify rows. (PDF pp. 11, 14.)

D - Incorrect: A combination can be a composite primary key, but an arbitrary combination is not sufficient. (PDF pp. 11, 14.)

Primary note proof

Study-note section FDB07

A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

A table can have several candidates and foreign keys but at most one declared primary-key constraint, possibly containing several columns.

Study-note section FDB16

PRIMARY KEY implies uniqueness and non-NULL.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Fundamental of Database Systems module | PDF page 14 | lines 17-20

Key Constraints- These are called uniqueness constraints since it ensures that every tuple in the relation should be unique. An attribute that can uniquely identify a tuple in a relation is called the key or primary key of the table. The value of the attribute for different tuples in the relation has to be unique. Primary keys must be unique and cannot be null.

Question sources

Computer Science MCQ practice collection · Question 132

Question 163 | AM0133

What is the purpose of foreign keys in a relational database?

A. To establish relationships between tables

B. To store encrypted data

C. To enforce data types in columns

D. To improve query performance

Hint

Match the validity requirement to the appropriate constraint definition.

Answer A: To establish relationships between tables

Foreign keys link child records to matching parent keys and preserve consistency between related tables.

Why each choice is right or wrong

A - Correct: Establishes the described reference relationship. (PDF pp. 48, 14.)

B - Incorrect: Encryption is a security mechanism, not the FK purpose. (PDF pp. 48, 14.)

C - Incorrect: Data types are controlled by domain constraints. (PDF pp. 48, 14.)

D - Incorrect: Performance improvement is not the stated FK purpose. (PDF pp. 48, 14.)

Primary note proof

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Study-note section FDB07

> A primary key is one selected candidate key; remaining candidates are alternate keys.

Study-note section FDB07

> Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys.

Related reading (context, not primary note proof)

Study-note section FDB38

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

> iv. Referential Integrity • A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key • Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Fundamental of Database Systems module | PDF page 14 | lines 21-26

> Referential Integrity Constraints - The Referential integrity constraints is specified between two relations or tables and used to maintain the consistency among the tuples in two relations. Referential Integrity constraints in DBMS are based on the concept of Foreign Keys. A foreign key is an important attribute of a relation which should be referred to in other relationships. Referential integrity constraint state happens where relation refers to a key attribute of a different or same relation. However, that key element must exist in the table.

Question sources

Computer Science MCQ practice collection · Question 133

Question 164 | AM0134

In the usual normalization sequence, which is the first normal form requiring every nonprime attribute to depend fully on every candidate key, rather than on a proper subset?

A. First normal form (1NF)

B. Second normal form (2NF)

C. Third normal form (3NF)

D. Boyce-Codd normal form (BCNF)

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer B: Second normal form (2NF)

Second normal form is the first stage requiring atomic relations to have no nonprime attribute partially dependent on any candidate key. First normal form addresses atomic values, so it does not impose that dependency rule. Third normal form and BCNF are stronger stages that inherit the absence of partial dependencies, but they are not the first stage introducing it. BCNF specifically requires every nontrivial functional-dependency determinant to be a superkey.

Why each choice is right or wrong

A - Incorrect: Incorrect: 1NF requires atomic values but does not exclude partial dependencies.

B - Correct: Correct: 2NF is the first stage excluding nonprime partial dependencies on candidate keys.

C - Incorrect: Incorrect: 3NF inherits 2NF and imposes its additional determinant/prime-attribute condition; it is not the first stage.

D - Incorrect: Incorrect: BCNF requires every nontrivial determinant to be a superkey; it is stronger than the first stage imposing the partial-dependency rule.

Primary note proof

Study-note section FDB20

> 1NF requires atomic values in the chosen relational domains and no repeating groups in cells.

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Study-note section FDB21

> For every nontrivial FD X→A, 3NF permits it when X is a superkey or A is prime. BCNF requires X to be a superkey. The mnemonic whole key and nothing but the key describes the common single-key case; it cannot replace the full tests when keys overlap. Remove dependencies of nonprime facts on nonkey determinants.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 10-15

> Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

> Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Computer Science MCQ practice collection · Question 134

Question 165 | AM0135

What defines a composite key in database design?

A. A key made up of two or more attributes

B. A foreign key that references multiple tables

C. A unique identifier for a column in a table

D. A key used for data encryption

Hint

Test uniqueness and minimality separately when classifying keys.

Answer A: A. A key made up of two or more attributes

Composite describes the number of attributes in a key: it contains two or more attributes. It does not mean the key has necessarily been selected as the primary key, nor does it refer to the number of tables reached by a reference. A column name alone is not a multicolumn identification key. Encryption uses a different meaning of key.

Why each choice is right or wrong

A - Correct: Two or more key attributes are the defining feature.

B - Incorrect: The number of referenced tables does not define composition.

C - Incorrect: A column identifier does not describe a multicolumn key.

D - Incorrect: An encryption key is a different concept.

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

> A composite-key example must include multiple columns; a single-column key is not composite.

Study-note section FDB07

A table can have several candidates and foreign keys but at most one declared primary-key constraint, possibly containing several columns.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 15 | lines 11-17

Partial Dependency - If an attribute which is not a member of the key is dependent on some part of the key (if we have composite/candidate key) then that attribute is partially functionally dependent on the key. Let {A, B} is the key set and C is no key attribute, then if {A, B} → C and [B → C or A → C] holds. We say C is partially functionally dependent on {A, B}. Partial dependency occurs when a non-prime attribute is functionally dependent on part of a candidate/composite key.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Question sources

Computer Science MCQ practice collection · Question 135

Question 166 | AM0136

What is the purpose of an index in a database?

A. To store large binary objects

B. To enforce referential integrity

C. To improve data retrieval performance

D. To encrypt sensitive data

Hint

Compare index organization with its access benefits and maintenance costs.

Answer C: To improve data retrieval performance

Indexes are auxiliary structures used to speed retrieval for search conditions, rather than stored binary values or security mechanisms.

Why each choice is right or wrong

A - Incorrect: BLOB is the named large-binary-object type. (PDF pp. 21, 19.)

B - Incorrect: Foreign keys handle referential links; indexes are defined for retrieval. (PDF pp. 21, 19.)

C - Correct: Exactly the stated retrieval-performance purpose. (PDF pp. 21, 19.)

D - Incorrect: Encryption is a separate security mechanism. (PDF pp. 21, 19.)

Primary note proof

Study-note section FDB25

> An index stores search keys with pointers or row identifiers. It often speeds selective retrieval but consumes space and must be maintained on writes. A full scan can win for a tiny table or a query returning most rows. Indexes do not themselves establish referential integrity.

Study-note section FDB23

> DATE represents a date and BLOB binary content such as images.

Study-note section FDB15

> Domain integrity restricts type and allowed values.

Related reading (context, not primary note proof)

Study-note section FDB38

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 21 | lines 5-9

> 5.5 Index Indexes are additional auxiliary access structures which are used to speed up the retrieval of records in response to certain search conditions. Index structures are additional files, which provide alternative ways to access the records without affecting the physical placement of records on disk.

Fundamental of Database Systems module | PDF page 19 | lines 11-15

> 5.2 Choosing Attribute Data Types CHAR - fixed-length character VARCHAR() - variable-length characters DATE - actual date BLOB - binary large object (good for graphics images, audio, sound clips, etc.)

Question sources

Computer Science MCQ practice collection · Question 136

Question 167 | AM0137

What is a data dictionary in database design?

A. A database table that stores user information

B. A collection of tables in a database

C. A file system used to store database files

D. A repository of metadata about the database structure

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer D: A repository of database metadata.

A data dictionary describes the database rather than serving as ordinary application data storage. FDB03 says it records definitions, types, constraints and often privileges and optimizer statistics. User information may appear in an application table, but that is not the dictionary definition. Nor is a dictionary simply all tables or the physical file system.

Why each choice is right or wrong

A - Incorrect: Ordinary user records are application data, not the defining purpose of a metadata repository.

B - Incorrect: A collection of tables is not specifically the repository describing those tables and constraints.

C - Incorrect: The physical file system stores files; a dictionary records descriptive database metadata.

D - Correct: The dictionary is the repository describing database structure, data types and constraints.

Primary note proof

Study-note section FDB03

> A schema describes names, structures and constraints; an instance, state or snapshot is the data at a particular time. Inserting a student changes the state, while adding a column changes the schema. Metadata is data describing data. The catalog or data dictionary records table definitions, types, constraints and often access privileges and optimizer statistics. A schema diagram draws tables and key references; labels such as database diagram can overlap.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 26-29

> Self-describing nature - A fundamental characteristic of the database approach is that the database system contains not only the database itself (data) but also a complete definition or description of the database structure and constraints. This information is called meta-data, and it describes the structure of the database.

Question sources

Computer Science MCQ practice collection · Question 137

Question 168 | AM0138

Which of the following is an example of a database constraint?

A. UNIQUE

B. SELECT

C. INSERT

D. DELETE

Hint

Read each SQL constraint's scope and enforcement rule.

Answer A: UNIQUE

Uniqueness constrains distinct tuple identifiers, whereas SELECT, INSERT and DELETE are language operations in the DML list. UNIQUE is the offered constraint name matching that property.

Why each choice is right or wrong

A - Correct: Uniqueness is explicitly an integrity condition. (PDF pp. 14, 26.)

B - Incorrect: SELECT is a retrieval command. (PDF pp. 14, 26.)

C - Incorrect: INSERT is a manipulation command. (PDF pp. 14, 26.)

D - Incorrect: DELETE is a manipulation command. (PDF pp. 14, 26.)

Primary note proof

Study-note section FDB16

> Example: CREATE TABLE Enrollment (StudentID INTEGER NOT NULL REFERENCES Student(StudentID), CourseID INTEGER NOT NULL REFERENCES Course(CourseID), Grade INTEGER CHECK (Grade BETWEEN 0 AND 100), PRIMARY KEY (StudentID,CourseID)); Each column declaration gives its name then type, with commas between declarations. CHECK is a constraint clause, not a standalone GRANT-like statement. POSITIVE is not a general SQL constraint keyword; CHECK (Amount > 0) expresses positivity.

Study-note section FDB26

> DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE.

Study-note section FDB26

> Some texts separate SELECT as DQL.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 14 | lines 17-20

> Key Constraints- These are called uniqueness constraints since it ensures that every tuple in the relation should be unique. An attribute that can uniquely identify a tuple in a relation is called

> the key or primary key of the table. The value of the attribute for different tuples in the relation has to be unique. Primary keys must be unique and cannot be null.

Fundamental of Database Systems module | PDF page 26 | lines 10-11

> 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Question sources

Computer Science MCQ practice collection · Question 138

Question 169 | AM0139

What is the purpose of referential integrity in a database?

A. To prevent unauthorized access to data

B. To ensure data consistency and integrity across related tables

C. To improve query performance

D. To encrypt sensitive data

Hint

Match the validity requirement to the appropriate constraint definition.

Answer B: To ensure data consistency and integrity across related tables

Referential integrity maintains consistency between related tuples by requiring non-null child references to resolve to existing parent keys. It controls valid relationships rather than authorization, performance, or encoding.

Why each choice is right or wrong

A - Incorrect: Unauthorized access is a security issue. (PDF pp. 14, 48.)

B - Correct: Matches the module's consistency-between-relations definition. (PDF pp. 14, 48.)

C - Incorrect: Query performance is not the referential validity rule. (PDF pp. 14, 48.)

D - Incorrect: Encryption is a separate security mechanism. (PDF pp. 14, 48.)

Primary note proof

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Study-note section FDB38

> Both encryption and access control can protect patient data.

Study-note section FDB38

> Authentication verifies identity; role-based authorization limits allowed records/actions; encryption protects data at rest or in transit; auditing records activity.

Related reading (context, not primary note proof)

Study-note section FDB25

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 14 | lines 21-26

> Referential Integrity Constraints - The Referential integrity constraints is specified between two relations or tables and used to maintain the consistency among the tuples in two relations. Referential Integrity constraints in DBMS are based on the concept of Foreign Keys. A foreign key is an important attribute of a relation which should be referred to in other relationships. Referential integrity constraint state happens where relation refers to a key attribute of a different or same relation. However, that key element must exist in the table.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

> iv. Referential Integrity • A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key • Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Variant 1 | Alternative version 1

What is the purpose of referential integrity in a database?

A. To prevent unauthorized access to data

B. To ensure data consistency and integrity

C. To improve query performance

D. To encrypt sensitive data

Answer B: To ensure data consistency and integrity

Referential integrity maintains consistency between related tuples by requiring non-null child references to resolve to existing parent keys. It controls valid relationships rather than authorization, performance, or encoding.

Why each choice is right or wrong

A - Incorrect: Authorization controls access; referential integrity checks reference validity (PDF pp.47-48).

B - Correct: Consistent valid references between related records match the integrity definition (PDF pp.14,48).

C - Incorrect: Retrieval speed is not the foreign-key consistency condition.

D - Incorrect: Encoding sensitive data is a separate security mechanism.

Primary note proof

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Study-note section FDB38

> Scalability, availability, fast retrieval, redundancy control and security are all reasonable design considerations.

Study-note section FDB38

> Authentication verifies identity; role-based authorization limits allowed records/actions; encryption protects data at rest or in transit; auditing records activity.

Related reading (context, not primary note proof)

Study-note section FDB25

Question sources

Computer Science MCQ practice collection · Question 139

Computer Science MCQ practice collection · Question 154

Question 170 | AM0140

What is denormalization in database design?

A. The process of breaking down a complex table into smaller tables

B. The process of adding redundancy to improve data retrieval performance

C. The process of eliminating duplicate values from a table

D. The process of converting data into a different format for storage

Hint

Compare design techniques by the specific problem each solves.

Answer B: Deliberately adding redundancy to improve selected retrieval workloads.

FDB38 defines denormalization as deliberately repeating or precomputing facts to reduce selected read costs, with extra maintenance work. It is a workload-driven trade-off rather than elimination of all duplication. Breaking tables apart typically describes normalization; changing storage format is a different concern. Thus B captures both the mechanism and its intended retrieval benefit.

Why each choice is right or wrong

A - Incorrect: Splitting tables according to dependencies is normalization, rather than the redundancy introduced by denormalization.

B - Correct: Denormalization deliberately repeats or precomputes information to improve selected retrieval workloads.

C - Incorrect: Eliminating duplication describes a different direction from deliberately introducing useful redundant facts.

D - Incorrect: Changing storage format does not define the dependency and redundancy trade-off of denormalization.

Primary note proof

Study-note section FDB38

> Normalization reduces dependency-driven anomalies. Indexing supplies search paths. Denormalization deliberately repeats or precomputes facts to reduce selected read costs, increasing maintenance work. Partitioning divides rows or columns; replication creates copies to improve locality or availability while complicating updates. These techniques can coexist. An e-commerce or gaming label alone does not select one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 17-27

> 4.4 Normalization Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. It is a technique of organizing the data in the database. It is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables. Normal forms Redundancy in a relation may causes insertion, deletion, and update anomalies. Normalization is database design technique that minimizes redundancy from a relation or a set of relations. Normalization rules divides larger tables into smaller tables and links them using relationship.

Question sources

Computer Science MCQ practice collection · Question 140

Question 171 | AM0141

What is the purpose of a view in a database?

A. To define the structure of a table

B. To store complex queries for reuse

C. To encrypt sensitive data

D. To enforce data types in columns

Hint

Distinguish a stored query definition from stored data and trace subquery scope.

Answer B: Store a query definition for reuse as a virtual table.

FDB33 says CREATE VIEW stores a query definition and presents a virtual table. B describes reusing that query, although a view need not be complex. Ordinary views do not inherently store result rows. Table definitions and column types belong to schema DDL, and views do not encrypt sensitive content merely by hiding columns or rows.

Why each choice is right or wrong

A - Incorrect: A view defines a virtual query result rather than the base table's storage structure.

B - Correct: A view saves a query definition that can be reused as a virtual table.

C - Incorrect: Restricting exposed data through a view does not encrypt that data.

D - Incorrect: Column data types are enforced by schema definitions and constraints rather than the view's primary purpose.

Primary note proof

Study-note section FDB33

> A subquery is a SELECT nested inside another statement. It may appear in WHERE, HAVING or FROM, not only the first two. IN tests membership; EXISTS tests whether at least one qualifying row exists. Correlated subqueries refer to the outer row. A scalar subquery must produce an appropriate single value. Beware NOT IN when its subquery can return NULL; NOT EXISTS often expresses the intended absence more safely.

Study-note section FDB33

> CREATE VIEW CSStudent AS SELECT StudentID,Name FROM Student WHERE Major = 'CS'; stores a query definition and presents a virtual table.

Study-note section FDB33

> An ordinary view does not necessarily store its result; a materialized view does.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 6 | lines 5-6

> Support of multiple views of the data - A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored.

Fundamental of Database Systems module | PDF page 56 | lines 7-12

> Views ● A subset of the database presented to some set of users ◦ SQL: CREATE VIEW viewname AS SELECT field1, field2, field3,..., FROM table1, table2 WHERE <where clause>; ◦ Note: —queries‖ in Access function as views

Fundamental of Database Systems module | PDF page 57 | lines 14-21

> ● Views, along with privileges, allow for customised access ● Views provide _derived‘ tables ◦ A view is the result of a SELECT statement which is treated like a table ◦ You can SELECT from (and

sometimes UPDATE etc) views just like tables Creating Views CREATE VIEW <name> AS <select stmt> ● <name> is the name of the new view

Question sources

Computer Science MCQ practice collection · Question 141

Question 172 | AM0142

Which database model represents data as a collection of objects?

A. Hierarchical model

B. Relational model

C. Object-oriented model

D. Network model

Hint

Compare the data models by how they organize records and relationships.

Answer C: Object-oriented model.

FDB02 says object models represent objects with identity, state and behavior. That matches a collection of objects directly. Hierarchical models use a parent-child tree, relational models use tables, and classic network models link records through owner/member pointers. Those representations should not be conflated merely because programming languages can wrap their data in objects.

Why each choice is right or wrong

A - Incorrect: The hierarchical model organizes records in a parent-child tree rather than primarily as objects.

B - Incorrect: The relational model represents data in tables with tuples and attributes.

C - Correct: The object model represents objects with identity, state and behavior.

D - Incorrect: The network model represents linked records through owner/member connections rather than object behavior.

Primary note proof

Study-note section FDB02

A data model supplies concepts for structure, relationships, constraints and operations. Conceptual ER models describe entities and relationships. Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents. Object models represent objects with identity, state and behavior. A physical model specifies record layouts and access paths. These are different representation choices, not stages of one query.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 8-19

> Types of Data Models 1. High-level or conceptual data models provide concepts that are close to the way many users perceive data. Entity - Relationship (E/R) model and Object-Oriented Model are examples of conceptual/high level data model. 2. Low-level or physical data models provide concepts that describe the details of how data is stored on the computer storage media. Concepts provided by physical data models are generally meant for computer specialists, not for end users. 3. representational (or implementation) data model provides concepts that may be easily understood by end users but that are not too far from the way data is organized in computer storage. Representational data models hide many details of data storage on disk but can be implemented on a computer system directly. The widely used relational data models are the network and hierarchical.

Fundamental of Database Systems module | PDF page 13 | lines 18-27

> 3.6 The Relational Database Model The relational model represents the database as a collection of relations. A table is represented with columns and rows. Each table is an entity in the ER diagram. Each row is known as a tuple. Each table of the column has a name or attribute of the entity. 3.7 Relational Model Concepts in DBMS Attribute: Each column in a Table. Attributes are the properties which define a relation Tables - In the Relational model, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes. Tuple - It is a single row of a table, which contains a single record.

Question sources

Computer Science MCQ practice collection · Question 142

Question 173 | AM0143

What is the purpose of data normalization in database design?

A. To improve data storage efficiency

B. To simplify the database structure

C. To eliminate data redundancy

D. To optimize query performance

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer C: To eliminate data redundancy

Normalization organizes/decomposes tables to avoid redundancy and anomalies. Storage may benefit, but the explicit defining purpose among these choices is reducing/eliminating repeated data.

Why each choice is right or wrong

A - Incorrect: Storage efficiency can be a consequence, but the cited definition does not guarantee a storage result. (PDF pp. 16.)

B - Incorrect: Decomposition does not necessarily simplify the whole schema. (PDF pp. 16.)

C - Correct: Eliminating redundancy is explicitly stated. (PDF pp. 16.)

D - Incorrect: Query speed is not the stated normalization goal. (PDF pp. 16.)

Primary note proof

Study-note section FDB19

> Suppose Enrollment repeats StudentName for every course. An update anomaly requires changing every copy of the student's name; missed copies disagree. An insertion anomaly prevents recording a new student until some enrollment exists. A deletion anomaly loses the only record of a student's details when their last enrollment is deleted. Revoke anomaly is not a normalization anomaly.

Study-note section FDB19

> Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design.

Study-note section FDB19

> Normalization separates facts according to their determinants, reducing avoidable repetition and associated anomalies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 17-29

> 4.4 Normalization Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. It is a technique of organizing the data in the database. It is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables. Normal forms Redundancy in a relation may causes insertion, deletion, and update anomalies. Normalization is database design technique that minimizes redundancy from a relation or a set of relations. Normalization rules divides larger tables into smaller tables and links them using relationship. Normal Forms are maintenance rules that are used to eliminate or reduce the redundancy in database relations. These normal forms are 1NF (First Normal Form), 2NF (Second Normal Form)

Question sources

Computer Science MCQ practice collection · Question 143

Question 174 | AM0143_2

In the customary single-key teaching case, which is the first normalization stage summarized as dependence on the key, the whole key, and nothing but the key?

A. First normal form (1NF)

B. Second normal form (2NF)

C. Third normal form (3NF)

D. Boyce-Codd normal form (BCNF)

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer C: Third normal form (3NF)

In the stated single-key teaching case, 2NF excludes partial dependencies but can still allow a nonprime fact to depend on the key through a nonkey determinant. The next defining stage is 3NF, summarized by the whole-key-and-nothing-but-the-key mnemonic. BCNF strengthens the determinant rule, and 4NF addresses multivalued dependencies. Stronger forms inherit earlier conditions, so the explicit first-stage scope makes 3NF the unique selection rather than treating inherited properties as rival answers.

Why each choice is right or wrong

A - Incorrect: Incorrect: 1NF concerns atomic values and does not eliminate the specified transitive dependencies.

B - Incorrect: Incorrect: 2NF excludes partial dependencies but can retain a nonprime transitive dependency through a nonkey determinant.

C - Correct: Correct: 3NF is the first stage imposing the specified nonprime transitive-dependency restriction in this teaching case.

D - Incorrect: Incorrect: BCNF strengthens 3NF by requiring every nontrivial determinant to be a superkey; it is not the first stage described.

Primary note proof

Study-note section FDB20

> 1NF requires atomic values in the chosen relational domains and no repeating groups in cells.

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Study-note section FDB21

> For every nontrivial FD X→A, 3NF permits it when X is a superkey or A is prime. BCNF requires X to be a superkey. The mnemonic whole key and nothing but the key describes the common single-key case; it cannot replace the full tests when keys overlap. Remove dependencies of nonprime facts on nonkey determinants.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 10-15

> Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

> Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Computer Science MCQ practice collection · Question 143

Question 175 | AM0144

Which database model represents data as a collection of records linked through pointers?

A. Hierarchical model

B. Relational model

C. Object-oriented model

D. Network model

Hint

Compare the data models by how they organize records and relationships.

Answer D: Network model.

FDB02 specifically describes the classic network model as records connected by pointer-based owner/member links, allowing multiple parents. That is the representation in the stem. Hierarchical data is restricted to a parent-child tree; relational data uses tables; object-oriented data emphasizes identity, state and behavior. Pointer-linked records therefore identify the network model in this comparison.

Why each choice is right or wrong

A - Incorrect: A hierarchical model restricts record organization to a parent-child tree rather than general network links.

B - Incorrect: A relational model uses tables and logical keys rather than the classic pointer-based network representation.

C - Incorrect: Object modeling is defined by identity, state and behavior, not this classic linked-record formulation.

D - Correct: Classic network databases connect records through pointer-based owner/member links.

Primary note proof

Study-note section FDB02

A data model supplies concepts for structure, relationships, constraints and operations. Conceptual ER models describe entities and relationships. Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents. Object models represent objects with identity, state and behavior. A physical model specifies record layouts and access paths. These are different representation choices, not stages of one query.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 8-19

Types of Data Models 1. High-level or conceptual data models provide concepts that are close to the way many users perceive data. Entity - Relationship (E/R) model and Object-Oriented Model are examples of conceptual/high level data model. 2. Low-level or physical data models provide concepts that describe the details of how data is stored on the computer storage media. Concepts provided by physical data models are generally meant for computer specialists, not for end users. 3. representational (or implementation) data model provides concepts that may be easily understood by end users but that are not too far from the way data is organized in computer storage. Representational data models hide many details of data storage on disk but can be implemented on a computer system directly. The widely used relational data models are the network and hierarchical.

Fundamental of Database Systems module | PDF page 13 | lines 18-27

3.6 The Relational Database Model The relational model represents the database as a collection of relations. A table is represented with columns and rows. Each table is an entity in the ER diagram. Each row is known as a tuple. Each table of the column has a name or attribute of the entity. 3.7 Relational Model Concepts in DBMS Attribute: Each column in a Table. Attributes are the properties which define a relation Tables - In the Relational model, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes. Tuple - It is a single row of a table, which contains a single record.

Question sources

Computer Science MCQ practice collection · Question 144

Question 176 | AM0145

What is the purpose of database indexing?

A. To encrypt sensitive data

B. To enforce data integrity constraints

C. To improve data retrieval performance

D. To ensure data consistency

Hint

Compare index organization with its access benefits and maintenance costs.

Answer C: To improve data retrieval performance

Indexes provide additional access paths and explicitly speed retrieval in response to search conditions; the purpose is performance of reads.

Why each choice is right or wrong

A - Incorrect: Encoding data is not the defined indexing purpose. (PDF pp. 21.)

B - Incorrect: The cited purpose is access speed, not integrity enforcement. (PDF pp. 21.)

C - Correct: Directly matches the retrieval definition. (PDF pp. 21.)

D - Incorrect: Consistency is a constraint/transaction concern rather than the indexing purpose defined. (PDF pp. 21.)

Primary note proof

Study-note section FDB25

> An index stores search keys with pointers or row identifiers. It often speeds selective retrieval but consumes space and must be maintained on writes. A full scan can win for a tiny table or a query returning most rows. Indexes do not themselves establish referential integrity.

Study-note section FDB25

> Secondary indexes provide other access paths and must represent all searchable values with record pointers or pointer lists.

Study-note section FDB38

> Indexing supplies search paths.

Related reading (context, not primary note proof)

Study-note section FDB15

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 21 | lines 5-9

5.5 Index Indexes are additional auxiliary access structures which are used to speed up the retrieval of records in response to certain search conditions. Index structures are additional files, which provide alternative ways to access the records without affecting the physical placement of records on disk.

Question sources

Computer Science MCQ practice collection · Question 145

Question 177 | AM0146

Which database model represents data as a collection of tables with rows and columns?

A. Hierarchical model

B. Relational model

C. Object-oriented model

D. Network model

Hint

Compare the data models by how they organize records and relationships.

Answer B: Relational model

The relational model represents data using relations, conventionally shown as tables of tuples and attributes. The other choices have different defining structures: a hierarchical model uses a parent-child tree, a classic network model uses pointer-based owner/member links, and an object model uses objects with identity, state and behavior. Products may combine features, but the question asks for the model defined by rows and columns, which is relational.

Why each choice is right or wrong

A - Incorrect: Incorrect: the hierarchical model organizes records in a parent-child tree.

B - Correct: Correct: the relational model uses relations represented as tables of rows and columns.

C - Incorrect: Incorrect: the object model represents objects with identity, state and behavior.

D - Incorrect: Incorrect: the classic network model uses pointer-based owner/member links, allowing multiple parents.

Primary note proof

Study-note section FDB02

A data model supplies concepts for structure, relationships, constraints and operations. Conceptual ER models describe entities and relationships. Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents. Object models represent objects with identity, state and behavior. A physical model specifies record layouts and access paths. These are different representation choices, not stages of one query.

Related reading (context, not primary note proof)

Study-note section FDB03

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 13 | lines 18-27

> 3.6 The Relational Database Model The relational model represents the database as a collection of relations. A table is represented with columns and rows. Each table is an entity in the ER diagram. Each row is known as a tuple. Each table of the column has a name or attribute of the entity. 3.7 Relational Model Concepts in DBMS Attribute: Each column in a Table. Attributes are the properties which define a relation Tables - In the Relational model, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes. Tuple - It is a single row of a table, which contains a single record.

Question sources

Computer Science MCQ practice collection · Question 146

Question 178 | AM0147

What is the purpose of a database schema?

A. To enforce data types in columns

B. To store database files

C. To define the structure and organization of a database

D. To ensure data integrity

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer C: To define the structure and organization of a database

A schema is the database description specified during design. It expresses structure, types, relationships and constraints; enforcing a particular constraint is narrower than the description's overall purpose.

Why each choice is right or wrong

A - Incorrect: Types are included, but enforcement alone is narrower than the schema description. (PDF pp. 7, 7.)

B - Incorrect: Files hold data physically; the schema describes the database. (PDF pp. 7, 7.)

C - Correct: Structure and organization match the schema definition. (PDF pp. 7, 7.)

D - Incorrect: Integrity conditions can be specified in a schema, but integrity alone is not its overall descriptive purpose. (PDF pp. 7, 7.)

Primary note proof

Study-note section FDB03

A schema describes names, structures and constraints; an instance, state or snapshot is the data at a particular time. Inserting a student changes the state, while adding a column changes the schema. Metadata is data describing data. The catalog or data dictionary records table definitions, types, constraints and often access privileges and optimizer statistics. A schema diagram draws tables and key references; labels such as database diagram can overlap.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 4-7

A data model a collection of concepts that can be used to describe the structure of a database and provides the necessary means to achieve data abstraction. By structure of a database we mean the data types, relationships, and constraints that apply to the data. Most data models also include a set of basic operations for specifying retrievals and updates on the database.

Fundamental of Database Systems module | PDF page 7 | lines 20-24

2.2 Schemas and Instances The description of a database is called the database schema, which is specified during database design and is not expected to change frequently. The data in the database at a particular moment of time is called a database state or snapshot. It is also called the current set of occurrences or instances in the database.

Question sources

Computer Science MCQ practice collection · Question 147

Question 179 | AM0148

What is the role of a database administrator (DBA) in database management?

A. To design and implement the database schema

B. To write complex SQL queries

C. To manage and maintain the database system

D. To develop database applications

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer C: To manage and maintain the database system

The DBA administers database and DBMS resources, authorizes access and coordinates/monitors use. Management and maintenance summarize those system responsibilities.

Why each choice is right or wrong

A - Incorrect: Schema design may be a duty in practice but is not the defining administrator role stated. (PDF pp. 6.)

B - Incorrect: Complex query writing is not the defining administrative responsibility. (PDF pp. 6.)

C - Correct: Matches resource administration and monitoring. (PDF pp. 6.)

D - Incorrect: Developing applications is a different role from the stated administration duties. (PDF pp. 6.)

Primary note proof

Study-note section FDB01

> The DBA manages access authorization, backup/recovery, availability, monitoring, tuning and routine maintenance, coordinating database use; designers define the schema and application developers build user-facing programs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 6 | lines 10-16

> 1.4 Actors on the scene The people whose jobs involve the day-to-day use of a large database are called actors on the scene. In a database environment, the primary resource is the database itself, and the secondary resource is the DBMS and related software. Administering these resources is the responsibility of the database administrator (DBA). The DBA is responsible for authorizing access to the database, coordinating and monitoring its use, and acquiring software and hardware resources as needed.

Question sources

Computer Science MCQ practice collection · Question 148

Computer Science MCQ practice collection · Question 169

Question 180 | AM0149

Which database model represents data as a hierarchical structure with parent-child relationships?

A. Hierarchical model

B. Relational model

C. Object-oriented model

D. Network model

Hint

Compare the data models by how they organize records and relationships.

Answer A: Hierarchical model.

The defining phrase is a hierarchical parent-child structure. FDB02 assigns that tree organization to the hierarchical model. The relational model uses tables; the object model represents identity, state and behavior; the network model permits more general links including multiple parents. A parent-child tree therefore selects A directly.

Why each choice is right or wrong

A - Correct: The hierarchical model organizes records into a parent-child tree.

B - Incorrect: The relational model represents data as tables rather than a prescribed parent-child tree.

C - Incorrect: The object model represents object identity, state and behavior rather than requiring a tree.

D - Incorrect: The network model permits more general connections and multiple parents rather than the restricted tree structure.

Primary note proof

Study-note section FDB02

> A data model supplies concepts for structure, relationships, constraints and operations. Conceptual ER models describe entities and relationships. Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents. Object models represent objects with identity, state and behavior. A physical model specifies record layouts and access paths. These are different representation choices, not stages of one query.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 8-19

> Types of Data Models 1. High-level or conceptual data models provide concepts that are close to the way many users perceive data. Entity - Relationship (E/R) model and Object-Oriented Model are examples of conceptual/high level data model. 2. Low-level or physical data models provide concepts that describe the details of how data is stored on the computer storage media. Concepts provided by physical data models are generally meant for computer specialists, not for end users. 3. representational (or implementation) data model provides concepts that may be easily understood by end users but that are not too far from the way data is organized in computer storage. Representational data models hide many details of data storage on disk but can be implemented on a computer system directly. The widely used relational data models are the network and hierarchical.

Fundamental of Database Systems module | PDF page 13 | lines 5-8

> Subclasses and super classes can be thought as useful ways to describe similarities in the entities represented. Because subclasses inherit their attributes and relationships from the superclass, the database designer can represent complex, hierarchical interdependencies (inheritance) within the system being described.

Question sources

Computer Science MCQ practice collection · Question 149

Question 181 | AM0150

What is the first step in the database design process?

A. Defining relationships between tables

B. Determining the database schema

C. Analyzing the information requirements

D. Creating database queries

Hint

Identify the design phase from its inputs and intended deliverable.

Answer C: Analyzing the information requirements

The initial design step is to understand the organization's information needs, required transactions, reports and constraints. Those requirements guide conceptual modeling, which is then mapped into a logical schema. Defining table relationships and determining the schema therefore depend on the earlier analysis. Queries express required operations against a model rather than replacing the initial work of identifying what information and behavior are needed. C is the first step in that sequence.

Why each choice is right or wrong

A - Incorrect: Incorrect: table relationships are designed to satisfy requirements already identified.

B - Incorrect: Incorrect: schema choices follow the analysis of required information and operations.

C - Correct: Correct: analyze information requirements before choosing the model and schema.

D - Incorrect: Incorrect: query construction depends on understanding the required information and intended database structure.

Primary note proof

Study-note section FDB05

> First collect business facts, required transactions, reports, volumes, privacy rules and quality targets. Conceptual design turns them into an ER/EER model. Logical design maps the conceptual model to relations, keys and constraints and checks normalization. Physical design chooses file organization, indexes and storage. Implementation creates and loads the actual objects; testing checks requirements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 7-20

3.2 Phases of database design The phases of database development life cycle (DDLC) in the Database Management System (DBMS) are explained below. Requirement Analysis The most important step in implementing a database system is to find out what is needed i.e what type of a database is required for the business organization, daily volume of data, how much data needs to be stored in the master files etc. In order to collect all this information, a database analyst spends a lot of time within the business organization talking to people, end users and getting acquainted with the day-to-day process. Conceptual Database Design In this phase the database designers will make a decision on the database model that perfectly suits the organization‘s requirement. The database designers will study the documents prepared by the analysis in the requirement analysis stage and then start development of a system model that fulfils the needs.

Question sources

Computer Science MCQ practice collection · Question 150

Question 182 | AM0151

Read pairs left→right. Student↔Course and Product↔Supplier permit many occurrences in both directions. Each employee belongs to one department with many employees. Each order belongs to one customer, who may place many orders. Which pair is one-to-many?

A. Student and Course

B. Employee and Department

C. Customer and Order

D. Product and Supplier

Hint

Read the relationship in both directions and check participation constraints.

Answer C: C. Customer and Order

Customer→Order is one-to-many because one customer can place several orders and each order belongs to one customer. Employee→Department runs in the opposite direction: many employees share one department, making it many-to-one. The student/course and product/supplier pairs explicitly permit many partners in both directions and are many-to-many. Cardinality follows these rules and the direction of the pair.

Why each choice is right or wrong

A - Incorrect: Student→Course is M:N under the stated enrollment rule.

B - Incorrect: Employee→Department is N:1.

C - Correct: Customer→Order is 1:N under the ownership rule.

D - Incorrect: Product→Supplier is M:N under the stated supply rule.

Primary note proof

Study-note section FDB09

> Cardinality specifies maximum associated occurrences: 1:1, 1:N or M:N. Minimum cardinality specifies optional/partial participation (zero permitted) or mandatory/total participation (at least one). A maximum of one does not mean participation is compulsory. Relationship degree counts participating entity roles: unary recursive, binary or ternary.

Study-note section FDB09

> If each employee belongs to one department and each department has many employees, Employee→Department is N:1; Department→Employee is 1:N.

Study-note section FDB09

> Read business rules in both directions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 24-31

> Cardinality The number of times an entity of an entity set participates in a relationship set is known as cardinality. Cardinality can be of different types: One to one (1-1) - When each entity in each entity set can take part only once in the relationship, the cardinality is one to one. One to many(1-M)- When entities in one entity set can take part only once in the relationship set and entities in other entity set can take part more than once in the relationship set, cardinality is many to one.

Fundamental of Database Systems module | PDF page 12 | lines 2-3

> Many to many (M-M) - When entities in all entity sets can take part more than once in the relationship cardinality is many to many.

Question sources

Computer Science MCQ practice collection · Question 151

Question 183 | AM0153

Which of the following is a benefit of using database indexes?

A. Increased data redundancy

B. Decreased data integrity

C. Improved data retrieval performance

D. Simplified data modeling

Hint

Compare index organization with its access benefits and maintenance costs.

Answer C: Improved data retrieval performance

The auxiliary index provides alternative access paths to speed retrieval under search conditions. This is the directly supported benefit.

Why each choice is right or wrong

A - Incorrect: Redundancy is not the benefit identified by the index definition. (PDF pp. 21.)

B - Incorrect: Reduced integrity is not a stated benefit. (PDF pp. 21.)

C - Correct: Retrieval improvement is explicitly stated. (PDF pp. 21.)

D - Incorrect: Simplification of the logical model is not the index's stated function. (PDF pp. 21.)

Primary note proof

Study-note section FDB25

> An index stores search keys with pointers or row identifiers. It often speeds selective retrieval but consumes space and must be maintained on writes. A full scan can win for a tiny table or a query returning most rows. Indexes do not themselves establish referential integrity.

Study-note section FDB25

> Secondary indexes provide other access paths and must represent all searchable values with record pointers or pointer lists.

Study-note section FDB25

> A dense index represents each search value; sparse indexes rely on ordered data between entries.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 21 | lines 5-9

> 5.5 Index Indexes are additional auxiliary access structures which are used to speed up the retrieval of records in response to certain search conditions. Index structures are additional files, which provide alternative ways to access the records without affecting the physical placement of records on disk.

Question sources

Computer Science MCQ practice collection · Question 153

Question 184 | AM0155

Employee has required, unique SSN and Employee ID values. Names and salaries may repeat. Which listed individual attributes are candidate keys? Select all that apply.

A. Social Security Number (SSN)

B. Employee Name

C. Employee ID

D. Employee Salary

Hint

Test uniqueness and minimality separately when classifying keys.

Answer A, C: A. Social Security Number (SSN); C. Employee ID

A candidate key is a minimal attribute set guaranteed to identify a tuple. Each required unique single-attribute identifier is already minimal, so SSN and Employee ID both qualify. Names and salaries may repeat and therefore cannot guarantee identification. The question asks for candidate keys rather than the one candidate later selected as the primary key, allowing both identifiers.

Why each choice is right or wrong

A - Correct: Required unique SSN identifies each employee and is minimal.

B - Incorrect: Repeated names are permitted, so Name is not a candidate key.

C - Correct: Required unique Employee ID independently identifies each employee.

D - Incorrect: Repeated salaries are permitted, so Salary is not a candidate key.

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

> If ID is unique, both {ID} and {ID,City} are superkeys, but only {ID} is minimal.

Study-note section FDB07

> Prefer stable, compact identifiers when practical, but Email, SSN or EmployeeID is a candidate only if business rules guarantee uniqueness.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Question sources

Computer Science MCQ practice collection · Question 155

Question 185 | AM0156

What is the purpose of a foreign key in database design?

A. To enforce data types in columns

B. To define primary key constraints

C. To establish relationships between tables

D. To optimize query performance

Hint

Match the validity requirement to the appropriate constraint definition.

Answer C: To establish relationships between tables

A foreign key links a child occurrence to the matching parent candidate/primary key. Its defining purpose is the intertable relationship.

Why each choice is right or wrong

A - Incorrect: Domain constraints specify legal types. (PDF pp. 48.)

B - Incorrect: The parent key is an identifying constraint, while the child FK expresses the reference. (PDF pp. 48.)

C - Correct: Matches the stated linking purpose. (PDF pp. 48.)

D - Incorrect: Performance is not the defining reference function. (PDF pp. 48.)

Primary note proof

Study-note section FDB15

Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Study-note section FDB07

A primary key is one selected candidate key; remaining candidates are alternate keys.

Study-note section FDB07

A composite key contains multiple attributes and need not be the chosen primary key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

> iv. Referential Integrity ● A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key ● Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Question sources

Computer Science MCQ practice collection · Question 156

Question 186 | AM0157

Which of the following database design principles promotes data independence?

A. Entity-Relationship (ER) modeling

B. Normalization

C. Database indexing

D. Data abstraction

Hint

Trace which schema levels a proposed change affects.

Answer D: Data abstraction

Abstraction hides irrelevant details and helps achieve data independence. Three-schema separation then permits changes at one level without forcing higher-level schema changes.

Why each choice is right or wrong

A - Incorrect: ER modeling describes conceptual structure but is not the explicit independence principle here. (PDF pp. 6, 8.)

B - Incorrect: Normalization addresses redundancy/anomalies. (PDF pp. 6, 8.)

C - Incorrect: Indexes address retrieval paths and performance. (PDF pp. 6, 8.)

D - Correct: Abstraction is explicitly connected to independence. (PDF pp. 6, 8.)

Primary note proof

Study-note section FDB04

> Physical data independence permits changing internal storage without changing the conceptual schema, for example adding an index. Logical data independence permits conceptual changes while preserving affected external views and application interfaces, for example splitting a table while maintaining a compatible view. It is harder because applications depend on logical meaning. Data abstraction hides irrelevant details; program-data insulation separates data definitions from application code.

Study-note section FDB05

> Conceptual design turns them into an ER/EER model.

Study-note section FDB05

> Logical design maps the conceptual model to relations, keys and constraints and checks normalization.

Related reading (context, not primary note proof)

Study-note section FDB19

Study-note section FDB25

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 6 | lines 2-4

> Data abstraction and Insulation between programs and data- Data Abstraction is a process of hiding unwanted or irrelevant details from the end user. It provides a different view and helps in achieving data independence which is used to enhance the security of data.

Fundamental of Database Systems module | PDF page 8 | lines 8-15

> 2.4 Data Independence The three-schema architecture can be used to further explain the concept of data independence, which can be defined as the capacity to change the schema at one level of a database system without having to change the schema at the next higher level. The two types of data independence: 1. Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs. 2. Physical data independence is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas do not need be changed as well.

Question sources

Computer Science MCQ practice collection · Question 157

Question 187 | AM0158

Which type of database relationship is represented by a join table?

A. One-to-one relationship

B. One-to-many relationship

C. Many-to-many relationship

D. Hierarchical relationship

Hint

Identify relationship degree and cardinality before choosing a relational mapping.

Answer C: Many-to-many relationship.

A join table stores pairs of references to the two participating entity tables, optionally with relationship attributes. FDB12 uses Enrollment(StudentID,CourseID,Grade) for the many-to-many Student-Course relationship. One-to-one and one-to-many relationships normally use a foreign key in an entity table instead. A hierarchy is a structural pattern, not the relationship cardinality represented by this usual association mapping.

Why each choice is right or wrong

A - Incorrect: A 1:1 relationship normally uses a UNIQUE foreign key rather than requiring a join table.

B - Incorrect: A 1:N relationship normally stores the one-side key in the many-side entity table.

C - Correct: A join or association table stores the pairs making up a many-to-many relationship.

D - Incorrect: Hierarchical organization is not the standard cardinality category represented by this association-table mapping.

Primary note proof

Study-note section FDB12

> A binary M:N relationship needs an association table in addition to the two entity tables. Enrollment(StudentID,CourseID,Grade) references Student and Course; (StudentID,CourseID) is a key if one enrollment per pair is allowed. Repeated attempts require Term or AttemptNo in the key. This ordinarily produces three tables.

Study-note section FDB11

> For 1:N, place the one-side key as a foreign key in the many-side table; store relationship attributes there.

Study-note section FDB11

> For 1:1, one side may hold the other key with a UNIQUE constraint.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 12 | lines 2-3

> Many to many (M-M) - When entities in all entity sets can take part more than once in the relationship cardinality is many to many.

Fundamental of Database Systems module | PDF page 12 | lines 8-24

> 3.4 Mapping ER-models to relational tables 1. For each strong entity type E, create a relation R that includes all the attributes of E. 2. Choose one of the key attributes of E as the primary key for R. For each weak entity type W with owner entity type E (which is a strong entity), create a relation R & include all attributes of ‗W‘ as attributes of R. Also, include the primary key attribute of the relation that correspond to the owner entity type as foreign key attributes of R . 3. For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. Foreign Key approach: Choose one of the relations-say S-and

include a foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S. 4. For each binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type. Include the primary key of the relation T as foreign key in S that represents the other entity type participating in R. Include any attributes of the 1:N relation type as attributes of S. 5. For each n-ary relationship type R, where n>2, create a new relationship S to represent R. Include the primary keys of the relations that represent the participating entity types as foreign key attributes in S. Also include any attributes of the n-ary relationship type as attributes of S.

Question sources

Computer Science MCQ practice collection · Question 158

Question 188 | AM0159

Employee ID uniquely identifies an employee and their name, salary and department. Names may repeat; the same salary may occur in different departments, and a department may contain different salaries. Which listed functional dependency is guaranteed?

A. Employee ID -> Employee Name

B. Employee Name -> Employee ID

C. Employee Salary -> Employee Department

D. Employee Department -> Employee Salary

Hint

Compute closures using the declared dependencies and test minimality.

Answer A: A. Employee ID -> Employee Name

A functional dependency is guaranteed by business rules for every legal state. Unique Employee ID determines all employee attributes, including Name, so A follows. Repeated names can correspond to different IDs, equal salaries can occur in different departments, and one department may contain different salaries. Those permissions prevent the other proposed directions from being guaranteed dependencies.

Why each choice is right or wrong

A - Correct: The unique employee identifier determines the employee's name.

B - Incorrect: Repeated names can belong to different IDs.

C - Incorrect: The same salary may occur in different departments.

D - Incorrect: One department may contain employees with different salaries.

Primary note proof

Study-note section FDB17

X→Y means every legal pair of tuples agreeing on X also agrees on Y. It is a business constraint, not a correlation observed in one sample. Full dependency requires all determinant attributes;

partial dependency holds through a proper subset. A prime attribute belongs to some candidate key; a nonprime attribute belongs to none.

Study-note section FDB17

Remove attributes and recompute to test minimality.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 15 | lines 5-9

A functional dependency is a constraint between two attributes or sets of attributes of entities in a database. A functional dependency is denoted by X → Y, between two sets of attributes X and Y that are subsets of relation R. The constraint is that, the values of Y component of a tuple in r depend on, or are determined by the values of X component. Alternatively, the values of X component of a tuple uniquely (or functionally) determine the values of Y component.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Question sources

Computer Science MCQ practice collection · Question 159

Question 189 | AM0160

What is the purpose of a data dictionary in database design?

A. To store user credentials

B. To manage database backups

C. To define the structure and attributes of data elements

D. To enforce referential integrity constraints

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer C: To describe the structure and attributes of data elements.

The dictionary contains metadata explaining table definitions, attribute types and constraints. FDB03 therefore supports C as its purpose. It may record privileges, but storing user credentials is not its defining function. Backup management is an administrative operation, and enforcing a reference requires DBMS constraint machinery rather than merely the repository that describes the constraint.

Why each choice is right or wrong

A - Incorrect: Credential storage is not the defining purpose of the database structure metadata repository.

B - Incorrect: Backup management is an administrative activity rather than the dictionary's structural-description purpose.

C - Correct: The dictionary records definitions, attributes, data types and constraints describing database elements.

D - Incorrect: It describes constraints; the DBMS enforcement machinery actually checks referential integrity during operations.

Primary note proof

Study-note section FDB03

> A schema describes names, structures and constraints; an instance, state or snapshot is the data at a particular time. Inserting a student changes the state, while adding a column changes the schema. Metadata is data describing data. The catalog or data dictionary records table definitions, types, constraints and often access privileges and optimizer statistics. A schema diagram draws tables and key references; labels such as database diagram can overlap.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 26-29

> Self-describing nature - A fundamental characteristic of the database approach is that the database system contains not only the database itself (data) but also a complete definition or description of the database structure and constraints. This information is called meta-data, and it describes the structure of the database.

Fundamental of Database Systems module | PDF page 7 | lines 4-7

> A data model a collection of concepts that can be used to describe the structure of a database and provides the necessary means to achieve data abstraction. By structure of a database we mean the data types, relationships, and constraints that apply to the data. Most data models also include a set of basic operations for specifying retrievals and updates on the database.

Question sources

Computer Science MCQ practice collection · Question 160

Computer Science MCQ practice collection · Question 184

Question 190 | AM0161

Compared with scanning all pages of a large heap file, which advantage can an RDBMS with a suitable index provide for a query selecting a small fraction of rows? Assume index lookup costs less than the scan.

A. Easy integration with external systems

B. Lower storage requirements

C. No need for data normalization

D. Faster data retrieval

Hint

Compare design techniques by the specific problem each solves.

Answer D: D. Faster data retrieval

The suitable index provides a search path for the selective query, reducing the work compared with scanning the large heap under the stated cost assumption. This supports faster retrieval for this workload. It does not remove normalization concerns, guarantee smaller storage or establish external-system compatibility. Index space and maintenance costs remain separate trade-offs from the query's retrieval advantage.

Why each choice is right or wrong

A - Incorrect: External integration is outside the stated lookup comparison.

B - Incorrect: Indexes consume additional storage rather than guaranteeing a reduction.

C - Incorrect: An index does not remove the need for dependency-aware design.

D - Correct: The stated cheaper indexed lookup supports faster retrieval for this query.

Primary note proof

Study-note section FDB38

> Normalization reduces dependency-driven anomalies. Indexing supplies search paths. Denormalization deliberately repeats or precomputes facts to reduce selected read costs, increasing maintenance work. Partitioning divides rows or columns; replication creates copies to improve locality or availability while complicating updates. These techniques can coexist. An e-commerce or gaming label alone does not select one.

Study-note section FDB25

> It often speeds selective retrieval but consumes space and must be maintained on writes.

Study-note section FDB25

> A full scan can win for a tiny table or a query returning most rows.

Related reading (context, not primary note proof)

Study-note section FDB01

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 14-21

> Database Management System (DBMS) is then a tool for creating and managing databases. DBMS is a software that facilities the processes of defining, constructing, manipulating, and sharing databases. Example: MySQL server, MS ACESS, Microsoft SQL server. 1.2 Database System versus File System The traditional file processing system is file-directory structure supported by a conventional operating system. A file system organization of data lucks a number of major features of a database system, such as Data redundancy and inconsistency, Difficulty in accessing data, Data isolation, Integrity problems, Atomicity problems, Concurrent access anomalies.

Fundamental of Database Systems module | PDF page 21 | lines 5-9

> 5.5 Index Indexes are additional auxiliary access structures which are used to speed up the retrieval of records in response to certain search conditions. Index structures are additional files, which provide alternative ways to access the records without affecting the physical placement of records on disk.

Question sources

Computer Science MCQ practice collection · Question 161

Question 191 | AM0163

OrderLine has one row per line number within an order. OrderID and LineNo together uniquely identify a row; neither alone does. Which listed attribute set is a composite key?

A. Employee ID

B. Product SKU

C. Customer Name

D. (OrderID, LineNo)

Hint

Test uniqueness and minimality separately when classifying keys.

Answer D: D. (OrderID, LineNo)

A composite key contains more than one attribute. OrderID identifies an order and LineNo identifies a line within it; the stated rules make their pair unique while neither component alone is sufficient. That establishes a minimal composite key. The other options each name only one attribute and therefore do not illustrate a composite key, regardless of any separate uniqueness policy.

Why each choice is right or wrong

A - Incorrect: Employee ID is a single attribute.

B - Incorrect: Product SKU is a single attribute.

C - Incorrect: Customer Name is a single listed attribute.

D - Correct: The pair is unique and requires both stated attributes.

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

> If ID is unique, both {ID} and {ID,City} are superkeys, but only {ID} is minimal.

Study-note section FDB07

> A composite-key example must include multiple columns; a single-column key is not composite.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 7-11

> Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Fundamental of Database Systems module | PDF page 15 | lines 11-17

> Partial Dependency - If an attribute which is not a member of the key is dependent on some part of the key (if we have composite/candidate key) then that attribute is partially functionally dependent on the key. Let {A, B} is the key set and C is no key attribute, then if {A, B} → C and [B → C or A → C] holds. We say C is partially functionally dependent on {A, B}. Partial dependency occurs when a non-prime attribute is functionally dependent on part of a candidate/composite key.

Question sources

Computer Science MCQ practice collection · Question 163

Question 192 | AM0164

What is the purpose of data validation in database design?

A. To enforce referential integrity constraints

B. To improve query performance

C. To eliminate data redundancy

D. To ensure data integrity and quality

Hint

Match the validity requirement to the appropriate constraint definition.

Answer D: To ensure data integrity and quality

Validation checks that input values meet legal domains/criteria, protecting integrity and consistency. D is the broad purpose; referential validation is one narrower kind rather than the whole definition.

Why each choice is right or wrong

A - Incorrect: Referential checks are one specific constraint, not the overall purpose of validation. (PDF pp. 47, 48, 49.)

B - Incorrect: Performance is not the legal-input goal. (PDF pp. 47, 48, 49.)

C - Incorrect: Redundancy reduction belongs to normalization. (PDF pp. 47, 48, 49.)

D - Correct: Integrity and input quality match domain validation and CHECK criteria. (PDF pp. 47, 48, 49.)

Primary note proof

Study-note section FDB15

> Domain integrity restricts type and allowed values. Key integrity enforces uniqueness. Entity integrity requires every primary-key component to be non-NULL. Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed. A parent need not have a child.

Study-note section FDB19

> Reduced redundancy can improve integrity and storage use, but extra keys and joins mean neither storage savings nor faster retrieval is guaranteed.

Study-note section FDB25

> Indexes do not themselves establish referential integrity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 47 | lines 4-19

> ● Security vs Integrity ◦ Database security makes sure that the user is authorised to access information ◦ Database integrity makes sure that (authorised) users use that information correctly ● Integrity constraints ◦ Domain constraints apply to data types ◦ Attribute constraints apply to columns ◦ Relation constraints apply to rows in a single table ◦ Database constraints apply between tables ● Intrarecord integrity (enforcing constraints on contents of fields, etc.) ● Referential Integrity (enforcing the validity of references between records in the database) ● Concurrency control (ensuring the validity of database updates in a shared multiuser

environment) ● The constraints we wish to impose in order to protect the database from becoming inconsistent.

Fundamental of Database Systems module | PDF page 48 | lines 4-8

ii. Attribute domain constraints ● Every attribute has a domain, that is a set of values that are legal for it to use ● For example: ◦ The domain of sex in the employee relation is —M‖ or —F‖ ● Domain ranges can be used to validate input to the database

Fundamental of Database Systems module | PDF page 49 | lines 19-22

● CREATE TABLE table-name (attr2 attr-type NOT NULL, forbids NULL values attrN attr-type CHECK (attrN = UPPER(attrN) verifies that the data meets certain criteria attrO attr-type DEFAULT default_value); Supplies default values

Question sources

Computer Science MCQ practice collection · Question 164

Question 193 | AM0165

Which normal forms have defining restrictions on functional dependencies? Select all that apply. Treat 1NF as the atomic-value/no-repeating-groups requirement.

A. First Normal Form (1NF)

B. Second Normal Form (2NF)

C. Third Normal Form (3NF)

D. Boyce-Codd Normal Form (BCNF)

Hint

Test determinants and prime attributes against the stated normal-form rules.

Answer B, C, D: B. Second Normal Form (2NF); C. Third Normal Form (3NF); D. Boyce-Codd Normal Form (BCNF)

Second normal form excludes nonprime partial functional dependencies on candidate keys. Third normal form checks each nontrivial dependency using the determinant's superkey status or the dependent's prime status. BCNF strengthens that test by requiring a superkey determinant. These three therefore use functional-dependency restrictions. First normal form instead establishes the atomic-value foundation and absence of repeating groups.

Why each choice is right or wrong

A - Incorrect: 1NF addresses atomic values and repeating groups rather than the stated dependency-based tests.

B - Correct: 2NF is defined through absence of nonprime partial functional dependencies on candidate keys.

C - Correct: 3NF tests nontrivial FDs using determinant superkeys and prime dependent attributes.

D - Correct: BCNF requires every nontrivial FD determinant to be a superkey.

Primary note proof

Study-note section FDB21

For every nontrivial FD X→A, 3NF permits it when X is a superkey or A is prime. BCNF requires X to be a superkey. The mnemonic whole key and nothing but the key describes the common single-key case; it cannot replace the full tests when keys overlap. Remove dependencies of nonprime facts on nonkey determinants.

Study-note section FDB20

A relation whose every candidate key is single-attribute has no nonprime partial-key dependency.

Study-note section FDB20

2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 2-15

First normal form (1NF) A relation (table) R is in 1NF if and only if (iff) all underlying domains of attributes contain only atomic (simple or indivisible) values. i.e. The value of any attribute in a tuple (row) must be a single value from the domain of their attribute. It is defined to disallow multivalued attributes, composite attributes, and their combinations. In other words, 1NF disallows relations within relations or relations as attribute values within tuples. It must be remove repeating groups (values) in a single field. The primary key with repeating group attributes are moved into a new table. When a relation contains no repeating groups (values), it is in first normal form (1NF). Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Computer Science MCQ practice collection · Question 165

Question 194 | AM0166

Which are direct representational purposes of conceptual/logical data modeling, rather than DBMS enforcement or physical query tuning? Select all that apply.

A. To enforce data constraints

B. To optimize query performance

C. To define the structure and organization of a database

D. To represent real-world entities and their relationships

Hint

Identify the design phase from its inputs and intended deliverable.

Answer C, D: C. To define the structure and organization of a database; D. To represent real-world entities and their relationships

Conceptual modeling represents domain entities and their relationships, while logical modeling defines database structures, keys and constraints. Both are direct representational purposes, so C and D qualify. Modeling can describe constraints, but the DBMS and application implementation enforce them during operations. Query performance can inform physical design, yet it is not the direct representational task specified in this question.

Why each choice is right or wrong

A - Incorrect: A model describes constraints, while implementation and DBMS machinery enforce them on actual operations.

B - Incorrect: Query performance can inform physical design, but it is not the general representational purpose asked here.

C - Correct: Modeling specifies the database's structure and organization.

D - Correct: Conceptual modeling represents real-world entities and their relationships.

Primary note proof

Study-note section FDB05

> First collect business facts, required transactions, reports, volumes, privacy rules and quality targets. Conceptual design turns them into an ER/EER model. Logical design maps the conceptual model to relations, keys and constraints and checks normalization. Physical design chooses file organization, indexes and storage. Implementation creates and loads the actual objects; testing checks requirements.

Study-note section FDB02

> A data model supplies concepts for structure, relationships, constraints and operations.

Study-note section FDB02

> Conceptual ER models describe entities and relationships.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 4-7

> A data model a collection of concepts that can be used to describe the structure of a database and provides the necessary means to achieve data abstraction. By structure of a database we mean the data types, relationships, and constraints that apply to the data. Most data models also include a set of basic operations for specifying retrievals and updates on the database.

Fundamental of Database Systems module | PDF page 9 | lines 3-6

> 3.1 Database Modeling High-level conceptual data models provide concepts for presenting data in ways that are close to the way people perceive data. A typical example is the entity relationship model, which uses main concepts like entities, attributes and relationships.

Fundamental of Database Systems module | PDF page 9 | lines 27-31

> 3.3 ERD (Entity Relationship Diagram) ERD a graphical technique for representing a database schema. It is a modeling tool used to depict graphically a DB design before it is actually implemented. It shows the various entities being modeled and the important relationships among them. ERD has three basic components Entity, Relationship and Attribute.

Question sources

Computer Science MCQ practice collection · Question 166

Question 195 | AM0167

Which option states a measurable nonfunctional quality requirement for a database system?

A. The database service must achieve at least 99.9% uptime.

B. User authentication

C. Entity-Relationship (ER) diagram

D. Data normalization

Hint

Compare design techniques by the specific problem each solves.

Answer A: A. The database service must achieve at least 99.9% uptime.

A nonfunctional requirement states a quality target such as availability or response time. The uptime threshold specifies a measurable availability requirement and fits that category. User authentication names system behavior, while an ER diagram is a modeling artifact and normalization is a design technique. These may support requirements, but they do not themselves state the quantified quality requested here.

Why each choice is right or wrong

A - Correct: The 99.9% uptime threshold is a measurable availability target.

B - Incorrect: Authentication names behavior rather than a quantified quality target.

C - Incorrect: An ER diagram is a design artifact.

D - Incorrect: Normalization is a design technique.

Primary note proof

Study-note section FDB38

Privacy requires a threat model. Authentication verifies identity; role-based authorization limits allowed records/actions; encryption protects data at rest or in transit; auditing records activity. Both encryption and access control can protect patient data. Scalability, availability, fast retrieval, redundancy control and security are all reasonable design considerations. Functional requirements describe behavior; nonfunctional requirements quantify qualities such as 99.9% uptime or response time. Authentication and encryption mechanisms alone do not uniquely establish that distinction.

Study-note section FDB19

Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design.

Study-note section FDB05

Conceptual design turns them into an ER/EER model.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 19 | lines 4-10

5.1 Introduction To translate the logical description of data into technical specifications for storing and retrieving data, a physical database design is needed. The goal is to create a design for storing and retrieving data that will provide adequate performance and insure database integrity, security, and availability. Some decisions need to be made on the physical database design. They are Attribute data types, data/file storages, File organizations and Indexes.

Fundamental of Database Systems module | PDF page 52 | lines 22-27

2.2 Database Security • Views or restricted subschemas • Authorization rules to identify users and the actions they can perform • User-defined procedures (and rule systems) to define additional constraints or limitations in using the database • Encryption to encode sensitive data

Fundamental of Database Systems module | PDF page 53 | lines 2-13

• Authentication schemes to positively identify a person attempting to gain access to the database • Database security is about controlling access to information • Some information should be available freely • Other information should only be available to certain people or groups • Many aspects to consider for security • Legal issues • Physical security • OS/Network security • Security policies and protocols • Encryption and passwords • DBMS security

Question sources

Computer Science MCQ practice collection · Question 167

Question 196 | AM0168

What is the purpose of data manipulation in database management?

A. To define the database schema

B. To create database backups

C. To insert, update, and delete data

D. To manage concurrent access to data

Hint

Classify each SQL command by the kind of change it performs.

Answer C: To insert, update, and delete data

The DML subsection includes INSERT, UPDATE and DELETE. SQL examples show insertion/deletion and an UPDATE privilege permits changing values, grounding the data-manipulation purpose. DML also includes retrieval, so C describes its modification part rather than all of DML.

Why each choice is right or wrong

A - Incorrect: Schema definition is DDL. (PDF pp. 26, 55, 77.)

B - Incorrect: Backup is not the DML purpose identified. (PDF pp. 26, 55, 77.)

C - Correct: Insertion, updating and deletion are explicitly DML operations. (PDF pp. 26, 55, 77.)

D - Incorrect: Concurrency control is a separate transaction-management responsibility. (PDF pp. 26, 55, 77.)

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 26 | lines 4-11

> SQL represents Structured Query Language. SQL is a comprehensive database language: It has statements for data definitions, and updates. Hence, it is both a DDL (Data Definition language) and a DML (Data Manipulation Language). 7.1.1 DDL (data definition language) Basic DDL commands we usually use are: CREATE, DROP, ALTER, TRUNCATE and RENAME 7.1.2 DML (data manipulation language) Basic DDL commands are: SELECT, PROJECT, JOIN, INSERT, DELETE, and UPDATE

Fundamental of Database Systems module | PDF page 55 | lines 2-5

> UPDATE(Salary) ON Employee TO Finance The user _Finance‘ can view the entire Employee table, and can change Salary values, but cannot change other values or pass on their privilege

Fundamental of Database Systems module | PDF page 77 | lines 14-24

> Example 1: Commit Transaction BEGIN TRANSACTION; DELETE FROM HumanResources.JobCandidate WHERE JobCandidateID = 13; COMMIT; Example 2: Rollback Transaction CREATE TABLE ValueTable (id int); BEGIN TRANSACTION; INSERT INTO ValueTable VALUES(1); INSERT INTO ValueTable VALUES(2); ROLLBACK;

Question sources

Computer Science MCQ practice collection · Question 168

Question 197 | AM0170

During conceptual database design, which listed goal directly concerns representing the business domain before choosing storage layouts and access paths?

A. Efficient storage of data

B. Secure data transmission

C. Optimal query performance

D. Effective representation of real-world entities

Hint

Identify the design phase from its inputs and intended deliverable.

Answer D: D. Effective representation of real-world entities

Conceptual design turns business facts into an ER/EER representation of entities and relationships. Effective representation of the domain is therefore the direct goal in this phase. Storage efficiency and query access paths are later physical design concerns. Secure transmission can be an important requirement, but it is not the conceptual representational task singled out by the revised question.

Why each choice is right or wrong

A - Incorrect: Storage layout efficiency belongs to physical decisions.

B - Incorrect: Transmission security is a requirement or mechanism, not the specified conceptual representation goal.

C - Incorrect: Query performance is addressed through later physical design and tuning.

D - Correct: Representing real-world entities directly matches conceptual design.

Primary note proof

Study-note section FDB05

First collect business facts, required transactions, reports, volumes, privacy rules and quality targets. Conceptual design turns them into an ER/EER model. Logical design maps the conceptual model to relations, keys and constraints and checks normalization. Physical design chooses file organization, indexes and storage. Implementation creates and loads the actual objects; testing checks requirements.

Study-note section FDB38

Indexing supplies search paths.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 16-26

Conceptual Database Design In this phase the database designers will make a decision on the database model that perfectly suits the organization‘s requirement. The database designers will study the documents prepared by the analysis in the requirement analysis stage and then start development of a system model that fulfils the needs. Logical database design (Data model mapping) The conceptual design is translated into internal model which includes mapping of all objects i.e design of tables, indexes, views, transaction, access privileges etc. Physical Design Physical design or Database implementation needs the formation of special storage related constructs. These constructs consist of storage groups, table spaces, data files, tables etc.

Fundamental of Database Systems module | PDF page 19 | lines 4-10

5.1 Introduction To translate the logical description of data into technical specifications for storing and retrieving data, a physical database design is needed. The goal is to create a design for storing and retrieving data that will provide adequate performance and insure database integrity, security, and availability. Some decisions need to be made on the physical database design. They are Attribute data types, data/file storages, File organizations and Indexes.

Question sources

Computer Science MCQ practice collection · Question 170

Question 198 | AM0171

Which are legitimate considerations when designing a database for a large e-commerce site? Select all that apply. Minimal redundancy means controlling avoidable duplication while permitting deliberate replicas.

A. High availability and scalability

B. Minimal data redundancy

C. Fast data retrieval

D. Secure user authentication

Hint

Compare design techniques by the specific problem each solves.

Answer A, B, C, D: A. High availability and scalability; B. Minimal data redundancy; C. Fast data retrieval; D. Secure user authentication

Large e-commerce databases need several complementary qualities. Availability and scalability support continued service and growth; retrieval speed supports interactive use; security protects accounts and records. Controlling avoidable redundancy reduces consistency and maintenance problems even where intentional replicas are useful. All listed considerations therefore belong in the design discussion, with their relative priority determined by the site's actual requirements.

Why each choice is right or wrong

A - Correct: High availability and scalability support growth and continued service for a large site.

B - Correct: Controlling unnecessary redundancy improves consistency and maintenance even when deliberate copies are retained.

C - Correct: Fast retrieval is a legitimate quality target for browsing, orders and customer-facing operations.

D - Correct: Secure authentication is a legitimate security consideration for customer and administrative access.

Primary note proof

Study-note section FDB38

> Privacy requires a threat model. Authentication verifies identity; role-based authorization limits allowed records/actions; encryption protects data at rest or in transit; auditing records activity. Both encryption and access control can protect patient data. Scalability, availability, fast retrieval, redundancy control and security are all reasonable design considerations. Functional requirements describe behavior; nonfunctional requirements quantify qualities such as 99.9% uptime or response time. Authentication and encryption mechanisms alone do not uniquely establish that distinction.

Study-note section FDB38

> In a CMS, tags/categories describe meaning, while foreign keys preserve valid references between content and tags.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 17-27

> 4.4 Normalization Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. It is a technique of organizing the data in the database. It is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables. Normal forms Redundancy in a relation may causes insertion, deletion, and update anomalies. Normalization is database design technique that minimizes redundancy from a relation or a set of relations. Normalization rules divides larger tables into smaller tables and links them using relationship.

Fundamental of Database Systems module | PDF page 19 | lines 4-10

> 5.1 Introduction To translate the logical description of data into technical specifications for storing and retrieving data, a physical database design is needed. The goal is to create a design for storing and retrieving data that will provide adequate performance and insure database integrity, security, and availability. Some decisions need to be made on the physical database design. They are Attribute data types, data/file storages, File organizations and Indexes.

Fundamental of Database Systems module | PDF page 21 | lines 5-9

> 5.5 Index Indexes are additional auxiliary access structures which are used to speed up the retrieval of records in response to certain search conditions. Index structures are additional files, which provide alternative ways to access the records without affecting the physical placement of records on disk.

Fundamental of Database Systems module | PDF page 53 | lines 14-21

> 2.2.1 DBMS Security Support ● DBMS can provide some security ◦ Each user has an account, username and password ◦ These are used to identify a user and control their access to information ● DBMS verifies password and checks a user‘s permissions when they try to ◦ Retrieve data ◦ Modify data ◦ Modify the database structure

Fundamental of Database Systems module | PDF page 116 | lines 19-28

> iv. Increased reliability and availability: • Reliability refers to system live time, that is, system is running efficiently most of the time. Availability is the probability that the system is continuously available (usable or accessible) during a time interval. • A distributed database system has multiple nodes (computers) and if one fails then others are available to do the job. v. Improved performance: • A distributed DBMS fragments the database to keep data closer to where it is needed most. • This reduces data management (access and modification) time significantly.

Fundamental of Database Systems module | PDF page 117 | lines 2-4

> vi. Easier expansion (scalability): • Allows new nodes (computers) to be added anytime without changing the entire configuration.

Question sources

Computer Science MCQ practice collection · Question 171

Question 199 | AM0172

Which listed mechanisms directly protect patient confidentiality, covering both protected stored/transmitted content and restrictions on authorized user actions? Select all that apply.

A. Database normalization

B. Data encryption

C. Role-based access control

D. Database indexing

Hint

Compare design techniques by the specific problem each solves.

Answer B, C: B. Data encryption; C. Role-based access control

Encryption protects patient content at rest or in transit against the disclosure threats it covers. Role-based access control restricts which users can read or change patient records. These are complementary privacy protections and both qualify. Normalization reduces dependency anomalies, while indexing supplies retrieval paths; neither mechanism by itself decides who may see patient information or protects encoded content.

Why each choice is right or wrong

A - Incorrect: Normalization reduces dependency anomalies but does not itself restrict unauthorized disclosure.

B - Correct: Encryption protects patient content at rest or in transit against appropriate disclosure threats.

C - Correct: Role-based access control limits which users may access patient records or perform actions.

D - Incorrect: Indexes provide search paths and do not by themselves enforce patient confidentiality.

Primary note proof

Study-note section FDB38

> Privacy requires a threat model. Authentication verifies identity; role-based authorization limits allowed records/actions; encryption protects data at rest or in transit; auditing records activity. Both encryption and access control can protect patient data. Scalability, availability, fast retrieval, redundancy control and security are all reasonable design considerations. Functional requirements describe behavior; nonfunctional requirements quantify qualities such as 99.9% uptime or response time. Authentication and encryption mechanisms alone do not uniquely establish that distinction.

Study-note section FDB25

> Secondary indexes provide other access paths and must represent all searchable values with record pointers or pointer lists.

Study-note section FDB19

> Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 52 | lines 22-27

2.2 Database Security ● Views or restricted subschemas ● Authorization rules to identify users and the actions they can perform ● User-defined procedures (and rule systems) to define additional constraints or limitations in using the database ● Encryption to encode sensitive data

Fundamental of Database Systems module | PDF page 53 | lines 2-18

● Authentication schemes to positively identify a person attempting to gain access to the database ● Database security is about controlling access to information ● Some information should be available freely ● Other information should only be available to certain people or groups ● Many aspects to consider for security ● Legal issues ● Physical security ● OS/Network security ● Security policies and protocols ● Encryption and passwords ● DBMS security 2.2.1 DBMS Security Support ● DBMS can provide some security ◦ Each user has an account, username and password ◦ These are used to identify a user and control their access to information ● DBMS verifies password and checks a user‘s permissions when they try to

Question sources

Computer Science MCQ practice collection · Question 172

Question 200 | AM0173

Which techniques can improve query performance for some suitable workloads, without implying that they always help? Select all that apply.

A. Indexing

B. Data replication

C. Denormalization

D. Partitioning

Hint

Compare design techniques by the specific problem each solves.

Answer A, B, C, D: A. Indexing; B. Data replication; C. Denormalization; D. Partitioning

Indexes add search paths for selective queries. Replicas can improve read locality, denormalization can reduce selected joins or computations, and partitioning can reduce relevant data access when aligned with queries. Each technique can therefore help some workloads. Their benefits are conditional and can involve extra storage or update costs, so selecting all four does not mean every database should use all of them.

Why each choice is right or wrong

A - Correct: Indexes can reduce reads for selective predicates when maintenance and lookup costs are justified.

B - Correct: Replication can place read copies near users or spread suitable read workloads.

C - Correct: Denormalization can avoid selected joins or repeated computation by storing redundant/precomputed facts.

D - Correct: Partitioning can limit relevant data or improve locality when partitioning matches query predicates.

Primary note proof

Study-note section FDB38

Normalization reduces dependency-driven anomalies. Indexing supplies search paths. Denormalization deliberately repeats or precomputes facts to reduce selected read costs, increasing maintenance work. Partitioning divides rows or columns; replication creates copies to improve locality or availability while complicating updates. These techniques can coexist. An e-commerce or gaming label alone does not select one.

Study-note section FDB25

B/B+ trees support ordered and range search; hashing targets equality; bitmap indexes suit some analytic workloads.

Study-note section FDB24

Choose organization from actual reads, writes and predicates.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 21 | lines 5-9

5.5 Index Indexes are additional auxiliary access structures which are used to speed up the retrieval of records in response to certain search conditions. Index structures are additional files, which provide alternative ways to access the records without affecting the physical placement of records on disk.

Fundamental of Database Systems module | PDF page 116 | lines 25-28

v. Improved performance: • A distributed DBMS fragments the database to keep data closer to where it is needed most. • This reduces data management (access and modification) time significantly.

Fundamental of Database Systems module | PDF page 117 | lines 18-29

I. Data Replication • The system maintain several identical copies of the relation & store each copy at a different site • In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead II. Data Fragmentation - Split a relation into logically related and correct parts. - The main reasons for fragmenting a relation are • Efficiency- data that is not needed by the local applications is not stored • Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency

Question sources

Computer Science MCQ practice collection · Question 173

Question 201 | AM0174

Users may follow many other users, and each user may be followed by many users. Which option describes the cardinality of this recursive user relationship, rather than a key constraint or procedural enforcement mechanism?

A. Primary key constraints

B. Foreign key constraints

C. Many-to-many relationships

D. Triggers and stored procedures

Hint

Identify relationship degree and cardinality before choosing a relational mapping.

Answer C: C. Many-to-many relationships

The follow rules permit multiple related users in both directions, which is many-to-many cardinality. A recursive association table can store two user references to represent those links. Primary and foreign keys support identification and referential integrity, but they are constraint mechanisms rather than the requested multiplicity classification. Triggers and stored procedures are also not a cardinality category.

Why each choice is right or wrong

A - Incorrect: Primary keys identify rows rather than name relationship multiplicity.

B - Incorrect: Foreign keys validate references rather than describe the many-to-many cardinality.

C - Correct: Both endpoints permit many partners, establishing M:N.

D - Incorrect: Procedures are implementation mechanisms rather than a cardinality type.

Primary note proof

Study-note section FDB12

> A recursive M:N friendship uses UserLink(UserID,OtherUserID), two foreign keys to User; decide whether links are directed and whether self-links are allowed. A recursive 1:N supervision uses a nullable SupervisorID reference in Employee. For a ternary Supply(SupplierID,PartID,ProjectID,Quantity), preserve all participating references; binary replacements can lose which triple occurred. Choose the association key from actual cardinalities rather than automatically assuming every reference is necessary.

Study-note section FDB07

> A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB15

> Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed.

Related reading (context, not primary note proof)

Study-note section FDB09

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 11 | lines 7-15

> Super Key − A set of attributes (one or more) that collectively identifies an entity in an entity set. Candidate Key − A minimal super key is called a candidate key. An entity set may have more than one candidate key. Primary Key − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set. Relationship A relationship type represents the association between entity types. For example, ‗Enrolled in' is a relationship type that exists between entity Student and Course. In ER diagram, relationship is represented by a diamond and connecting the entities with lines.

Fundamental of Database Systems module | PDF page 12 | lines 2-3

> Many to many (M-M) - When entities in all entity sets can take part more than once in the relationship cardinality is many to many.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

> iv. Referential Integrity • A ―foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key • Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Question sources

Computer Science MCQ practice collection · Question 174

Question 202 | AM0175

In the usual normalization sequence, which is the first normal form excluding transitive dependence of nonprime attributes on a key through a nonkey determinant?

A. First Normal Form (1NF)

B. Second Normal Form (2NF)

C. Third Normal Form (3NF)

D. Boyce-Codd Normal Form (BCNF)

Hint

Check atomicity, candidate keys and partial dependencies in that order.

Answer C: Third Normal Form (3NF)

In the stated single-key teaching case, 2NF excludes partial dependencies but can still allow a nonprime fact to depend on the key through a nonkey determinant. The next defining stage is 3NF, summarized by the whole-key-and-nothing-but-the-key mnemonic. BCNF strengthens the determinant rule, and 4NF addresses multivalued dependencies. Stronger forms inherit earlier conditions, so the explicit first-stage scope makes 3NF the unique selection rather than treating inherited properties as rival answers.

Why each choice is right or wrong

A - Incorrect: Incorrect: 1NF concerns atomic values and does not eliminate the specified transitive dependencies.

B - Incorrect: Incorrect: 2NF excludes partial dependencies but can retain a nonprime transitive dependency through a nonkey determinant.

C - Correct: Correct: 3NF is the first stage imposing the specified nonprime transitive-dependency restriction in this teaching case.

D - Incorrect: Incorrect: BCNF strengthens 3NF by requiring every nontrivial determinant to be a superkey; it is not the first stage described.

Primary note proof

Study-note section FDB20

> 1NF requires atomic values in the chosen relational domains and no repeating groups in cells.

Study-note section FDB20

> 2NF requires 1NF and no nonprime attribute partially dependent on any candidate key.

Study-note section FDB21

> For every nontrivial FD X→A, 3NF permits it when X is a superkey or A is prime. BCNF requires X to be a superkey. The mnemonic whole key and nothing but the key describes the common single-key case; it cannot replace the full tests when keys overlap. Remove dependencies of nonprime facts on nonkey determinants.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 17 | lines 10-15

> Second normal form (2NF) Second normal form (2NF) is based on the concept of full functional dependency. A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on the primary key (i.e. the non-prime attributes are not partially dependent on key attributes). Remove any partially dependent attributes and place them in another relation. A partial dependency is when the attribute(s) is/are dependent on a part of a key.

Fundamental of Database Systems module | PDF page 18 | lines 5-8

Third Normal Form (3NF) A relation schema R is in third normal form (3NF) if it is in 2NF and every non-prime attribute of _R‘ are fully functionally dependent on every key of R and non-transitively dependent on every key of R.

Question sources

Computer Science MCQ practice collection · Question 175

Question 203 | AM0176

What is the purpose of denormalization in database design for real-world scenarios?

A. To eliminate data redundancy

B. To optimize query performance

C. To enforce referential integrity

D. To simplify the database schema

Hint

Compare design techniques by the specific problem each solves.

Answer B: To improve selected query performance.

Denormalization repeats or precomputes facts so that selected queries can perform less work, such as fewer joins. FDB38 also states the cost: more maintenance when underlying facts change. It does not aim to eliminate redundancy or enforce foreign-key references. A simpler query path may result, but a simpler schema is not the essential purpose.

Why each choice is right or wrong

A - Incorrect: Denormalization introduces controlled redundancy instead of eliminating it.

B - Correct: Its goal is to reduce costs of selected read queries using repeated or precomputed information.

C - Incorrect: Foreign-key constraints enforce references; denormalization itself does not establish referential integrity.

D - Incorrect: It may simplify selected reads, but schema simplicity is not the defining or guaranteed goal.

Primary note proof

Study-note section FDB38

Normalization reduces dependency-driven anomalies. Indexing supplies search paths. Denormalization deliberately repeats or precomputes facts to reduce selected read costs, increasing maintenance work. Partitioning divides rows or columns; replication creates copies to improve locality or availability while complicating updates. These techniques can coexist. An e-commerce or gaming label alone does not select one.

Study-note section FDB15

Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed.

Study-note section FDB15

Key integrity enforces uniqueness.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 17-27

4.4 Normalization Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. It is a technique of organizing the data in the database. It is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables. Normal forms Redundancy in a relation may causes insertion, deletion, and update anomalies. Normalization is database design technique that minimizes redundancy from a relation or a set of relations. Normalization rules divides larger tables into smaller tables and links them using relationship.

Question sources

Computer Science MCQ practice collection · Question 176

Question 204 | AM0178

Which listed modeling technique directly includes superclass/subclass inheritance? Interpret ER modeling in A as basic ER without the EER extension, relational modeling as ordinary tables/keys, and XML modeling as document structure.

A. Entity-Relationship (ER) modeling

B. Object-oriented data modeling

C. Relational data modeling

D. XML data modeling

Hint

Review inheritance, specialization and category constraints.

Answer B: B. Object-oriented data modeling

Object-oriented data modeling directly represents superclass/subclass inheritance. Enhanced ER also supports inheritance, but A is explicitly limited to basic ER in this question. Ordinary relational tables can encode subtype data through references without making inheritance a direct modeling construct, and document structure is not the object inheritance mechanism being tested. The scope distinction makes the selected technique definite.

Why each choice is right or wrong

A - Incorrect: Basic ER excludes the EER inheritance extension under the stated scope.

B - Correct: Object models directly support superclass/subclass inheritance.

C - Incorrect: Ordinary relational tables encode facts and references rather than directly declaring this inheritance construct.

D - Incorrect: Document structure is not the direct object inheritance model asked for.

Primary note proof

Study-note section FDB13

> EER extends ER with superclass/subclass structure. A subclass inherits the superclass attributes and relationships. Specialization divides Student into Undergraduate and Graduate; generalization combines common properties into Student. Disjoint subclasses cannot overlap; overlapping subclasses can. Total specialization requires every superclass member in some subclass; partial specialization does not.

Study-note section FDB13

> Object models also support inheritance; distinguish basic ER notation from EER when comparing models.

Study-note section FDB13

> Structural EER inheritance alone does not define executable object methods.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 8-11

> Types of Data Models 1. High-level or conceptual data models provide concepts that are close to the way many users perceive data. Entity - Relationship (E/R) model and Object-Oriented Model are examples of conceptual/high level data model.

Fundamental of Database Systems module | PDF page 12 | lines 25-31

> 3.5 Enhanced Entity Relationship (EER) Model The complexityof the data is increasing so it becomes more and more difficult to use the traditional ER model for database modeling. To reduce this complexity of modeling we have to make improvements or enhancements to the existing ER model to make it able to handle the complex application in a better way. Enhanced ER (EER) model is a database model that incorporates the concepts of class/subclass relationships and type inheritance into the ER model. The EER model includes all the modeling

Fundamental of Database Systems module | PDF page 13 | lines 2-8

> concepts of the ER model. In addition, it includes the concepts of subclass and superclass, categories (UNION types), attribute and relationship inheritance. EER is created to design more accurate database schemas. Subclasses and super classes can be thought as useful ways to describe similarities in the entities represented. Because subclasses inherit their attributes and

relationships from the superclass, the database designer can represent complex, hierarchical interdependencies (inheritance) within the system being described.

Question sources

Computer Science MCQ practice collection · Question 178

Question 205 | AM0179

Which CMS features directly support either semantic grouping of content or valid stored references between content and tags? Select all that apply.

A. Referential integrity constraints

B. Full-text search capabilities

C. Content categorization and tagging

D. Distributed database architecture

Hint

Compare design techniques by the specific problem each solves.

Answer A, C: A. Referential integrity constraints; C. Content categorization and tagging

Categories and tags describe semantic associations among content items. Referential integrity constraints keep the stored links between content and tags valid. These two features address meaning and reference validity respectively, so both fit. Full-text search supports finding text, while a distributed architecture concerns data placement and coordination. Neither is the requested direct representation or validation of content-to-tag relationships.

Why each choice is right or wrong

A - Correct: Referential constraints preserve valid links between content and related rows, such as tags.

B - Incorrect: Full-text search finds textual content rather than primarily defining or validating item relationships.

C - Correct: Categories and tags describe semantic groupings and associations among content items.

D - Incorrect: Distribution determines placement and coordination rather than directly representing the stated content relationships.

Primary note proof

Study-note section FDB38

In a CMS, tags/categories describe meaning, while foreign keys preserve valid references between content and tags. A ContentTag(ContentID,TagID) association models multiple tags per item; these functions complement each other.

Study-note section FDB15

Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed.

Study-note section FDB15

Key integrity enforces uniqueness.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 27-31

3.3 ERD (Entity Relationship Diagram) ERD a graphical technique for representing a database schema. It is a modeling tool used to depict graphically a DB design before it is actually implemented. It shows the various entities being modeled and the important relationships among them. ERD has three basic components Entity, Relationship and Attribute.

Fundamental of Database Systems module | PDF page 48 | lines 13-17

iv. Referential Integrity ● A —foreign key‖ links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key ● Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity

Fundamental of Database Systems module | PDF page 116 | lines 25-28

v. Improved performance: • A distributed DBMS fragments the database to keep data closer to where it is needed most. • This reduces data management (access and modification) time significantly.

Question sources

Computer Science MCQ practice collection · Question 179

Question 206 | AM0180

Which database design technique is used to represent a hierarchical structure of data elements?

A. Relational data modeling

B. Network data modeling

C. Hierarchical data modeling

D. Object-relational data modeling

Hint

Compare the data models by how they organize records and relationships.

Answer C: Hierarchical data modeling.

A hierarchical structure organizes elements into parent-child levels. FDB02 assigns this representation to the hierarchical model, so C directly matches the requested design concept. Relational modeling instead uses tables, network modeling permits more general record links, and richer object-relational types do not by themselves require a hierarchical organization.

Why each choice is right or wrong

A - Incorrect: Relational modeling represents facts in tables rather than requiring a hierarchical tree.

B - Incorrect: Network modeling permits linked records with multiple parents rather than a restricted hierarchy.

C - Correct: Hierarchical modeling explicitly represents the parent-child structure asked for in the stem.

D - Incorrect: An object-relational approach adds richer types to relational storage rather than defining this tree organization.

Primary note proof

Study-note section FDB02

> A data model supplies concepts for structure, relationships, constraints and operations. Conceptual ER models describe entities and relationships. Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents. Object models represent objects with identity, state and behavior. A physical model specifies record layouts and access paths. These are different representation choices, not stages of one query.

Study-note section FDB-S02

> Richer relational types do not by themselves require all data to form a parent-child tree; that representation is the defining concern of hierarchical modeling.

Study-note section FDB-S02

> An object-relational DBMS retains relational data management while supporting richer extensions such as user-defined data types, functions and operators.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 8-19

> Types of Data Models 1. High-level or conceptual data models provide concepts that are close to the way many users perceive data. Entity - Relationship (E/R) model and Object-Oriented Model are examples of conceptual/high level data model. 2. Low-level or physical data models provide concepts that describe the details of how data is stored on the computer storage media. Concepts provided by physical data models are generally meant for computer specialists, not for end users. 3. representational (or implementation) data model provides concepts that may be easily understood by end users but that are not too far from the way data is organized in computer

> storage. Representational data models hide many details of data storage on disk but can be implemented on a computer system directly. The widely used relational data models are the network and hierarchical.

Fundamental of Database Systems module | PDF page 13 | lines 5-8

> Subclasses and super classes can be thought as useful ways to describe similarities in the entities represented. Because subclasses inherit their attributes and relationships from the superclass, the database designer can represent complex, hierarchical interdependencies (inheritance) within the system being described.

Question sources

Computer Science MCQ practice collection · Question 180

Question 207 | AM0181

Which benefit directly follows from reducing dependency-driven insertion, deletion and update anomalies through normalization, without assuming reduced storage or simpler queries?

A. Simplified data retrieval

B. Increased data redundancy

C. Enhanced data integrity

D. Reduced storage requirements

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer C: C. Enhanced data integrity

Normalization separates facts according to their determinants, reducing repeated facts and the insertion, deletion and update anomalies they cause. That directly supports enhanced data integrity. Storage savings and simpler retrieval are not guaranteed because decomposition can introduce additional keys and joins. Increased redundancy is opposite to the intended reduction of avoidable repetition, so it is not the benefit identified by this question.

Why each choice is right or wrong

A - Incorrect: Normalized retrieval may require more joins, so simplification does not follow.

B - Incorrect: Normalization aims to reduce avoidable repetition.

C - Correct: Reducing inconsistent updates and related anomalies improves integrity.

D - Incorrect: Storage savings can be offset by extra keys and relation overhead.

Primary note proof

Study-note section FDB19

> Suppose Enrollment repeats StudentName for every course. An update anomaly requires changing every copy of the student's name; missed copies disagree. An insertion anomaly

prevents recording a new student until some enrollment exists. A deletion anomaly loses the only record of a student's details when their last enrollment is deleted. Revoke anomaly is not a normalization anomaly.

Study-note section FDB19

Normalization separates facts according to their determinants, reducing avoidable repetition and associated anomalies.

Study-note section FDB19

Reduced redundancy can improve integrity and storage use, but extra keys and joins mean neither storage savings nor faster retrieval is guaranteed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 16 | lines 5-29

4.3 Database Anomalies Database Anomalies are the flaws in databases which occurs because of poor database design and storing everything in a single flat database. They are problems that are caused usually by redundant data. Types of database anomalies are Insertion anomaly, Deletion anomaly, and Update anomaly. Insertion Anomaly - occurs when the entire primary key is not known and the database cannot insert a new record properly. This would violate entity integrity. It can be avoided by using a sequence number for the primary key. Deletion Anomaly - happens when a record is deleted that results in the loss of other related data. It can be avoided by normalizing tables in a database to minimize their dependency. Update Anomaly - occurs when a change to one attribute value causes the database to either contain inconsistent data or causes multiple records to need changing. It may be prevented by making sure tables are in third normal form. 4.4 Normalization Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. It is a technique of organizing the data in the database. It is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables. Normal forms Redundancy in a relation may causes insertion, deletion, and update anomalies. Normalization is database design technique that minimizes redundancy from a relation or a set of relations. Normalization rules divides larger tables into smaller tables and links them using relationship. Normal Forms are maintenance rules that are used to eliminate or reduce the redundancy in database relations. These normal forms are 1NF (First Normal Form), 2NF (Second Normal Form)

Question sources

Computer Science MCQ practice collection · Question 181

Question 208 | AM0182

Which database design concept allows the division of a large database into smaller, manageable units?

A. Data partitioning

B. Database replication

C. Data migration

D. Data archiving

Hint

Distinguish logical records, disk pages and physical file organization.

Answer A: Data partitioning.

FDB24 explicitly contrasts partitioning, replication, migration and archiving. Partitioning divides the dataset into smaller portions, matching the question. Replication makes additional copies, migration moves data, and archiving takes older data out of active service. Those operations can accompany partitioning, but none of them defines division into manageable units.

Why each choice is right or wrong

A - Correct: Partitioning divides a large dataset into smaller portions or manageable units.

B - Incorrect: Replication creates copies rather than dividing the original dataset into distinct units.

C - Incorrect: Migration moves data to another location or system rather than defining a split.

D - Incorrect: Archiving removes older data from active service rather than generally dividing the database.

Primary note proof

Study-note section FDB24

> A page/block is a unit of database I/O and holds several records. For fixed unspanned records of size r in usable page space B, the blocking factor is floor(B/r), and n records occupy ceil(n/blocking factor) pages. Page headers and slots reduce usable space.

Study-note section FDB24

> Partitioning splits a large dataset; replication copies it; migration moves it; archiving removes older data from active service.

Study-note section FDB24

> A heap has no search-key order and supports simple inserts, but searches may scan all pages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 117 | lines 15-30

6.2 Data Replication and Fragmentation: Distributed data storage • There are two approaches to store the relation in the distributed database : Replication and Fragmentation I. Data Replication • The system maintain several identical copies of the relation & store each copy at a different site • In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead II. Data Fragmentation - Split a relation into logically related and correct parts. - The main reasons for fragmenting a relation are • Efficiency- data that is not needed by the local applications is not stored • Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency - but reconstruction of the whole relation will require accessing data from all sites

Fundamental of Database Systems module | PDF page 118 | lines 2-18

• A relation can be fragmented in two ways: • Horizontal fragmentation • It is a horizontal subset of a relation which contain those of rows which satisfy selection conditions. • Consider the Employee relation with selection condition (DNO = 5). All rows satisfy this condition will create a subset which will be a horizontal fragment of Employee relation. • A selection condition may be composed of several conditions connected by AND or OR. • Vertical fragmentation • It is a subset of a relation which is created by a subset of columns. Thus a vertical fragment of a relation will contain values of selected columns. - Consider the Employee relation. A vertical fragment of can be created by keeping the values of Name, Bdate, Sex, and Address. - Because there is no condition for creating a vertical fragment, each fragment must include the primary key attribute of the parent relation Employee. In this way all vertical fragments of a relation are connected.

Question sources

Computer Science MCQ practice collection · Question 182

Question 209 | AM0183

Which database design technique is used to capture the relationships between entities in a graphical representation?

A. Relational data modeling

B. Entity-Relationship (ER) modeling

C. Object-oriented data modeling

D. Document data modeling

Hint

Review entity, attribute and relationship notation before interpreting the model.

Answer B: Entity-Relationship (ER) modeling

ERD is explicitly a graphical database-schema technique showing modeled entities and their important relationships. This directly establishes ER modeling for the graphical representation asked about.

Why each choice is right or wrong

A - Incorrect: Relational modeling describes tables; ERD is the named graphical technique. (PDF pp. 9.)

B - Correct: ER modeling is the explicit graphical entities/relationships technique. (PDF pp. 9.)

C - Incorrect: Object-oriented is named but not the described ERD technique. (PDF pp. 9.)

D - Incorrect: Document modeling is not the technique defined in the module. (PDF pp. 9.)

Primary note proof

Study-note section FDB06

> An entity is a distinguishable real or conceptual thing, such as one student or one course. An entity type describes common properties; its current collection of entities is an entity set. A relationship associates entities. In Chen notation rectangles represent entity types, diamonds relationships and ellipses attributes; double ellipses denote multivalued attributes and dashed ellipses derived attributes. Always inspect the diagram legend.

Study-note section FDB02

> Conceptual ER models describe entities and relationships.

Study-note section FDB02

> Relational models use tables; hierarchical models organize records in a parent-child tree; classic network models connect records through pointer-based owner/member links, allowing multiple parents.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 9 | lines 27-31

> 3.3 ERD (Entity Relationship Diagram) ERD a graphical technique for representing a database schema. It is a modeling tool used to depict graphically a DB design before it is actually implemented. It shows the various entities being modeled and the important relationships among them. ERD has three basic components Entity, Relationship and Attribute.

Question sources

Computer Science MCQ practice collection · Question 183

Question 210 | CS0049

In contrary to the traditional file-based approach, in the database approach:

A. Programs define and access their own data

B. Store data items in multiple copies

C. Don't care about the definition of programs and data

D. Data and programs are defined independently

Hint

Trace which schema levels a proposed change affects.

Answer D: Data and programs are defined independently

The database approach insulates programs from data definitions and supports data independence. This is the relevant distinction from programs managing their own files.

Why each choice is right or wrong

A - Incorrect: Own program-specific definitions do not express the database approach's insulation. (PDF pp. 5, 6, 8.)

B - Incorrect: Multiple uncontrolled copies are not the defining benefit and can cause inconsistency. (PDF pp. 5, 6, 8.)

C - Incorrect: The approach explicitly maintains metadata and definitions, not indifference to them. (PDF pp. 5, 6, 8.)

D - Correct: Independent definitions match program/data insulation and independence. (PDF pp. 5, 6, 8.)

Primary note proof

Study-note section FDB04

> Physical data independence permits changing internal storage without changing the conceptual schema, for example adding an index. Logical data independence permits conceptual changes while preserving affected external views and application interfaces, for example splitting a table while maintaining a compatible view. It is harder because applications depend on logical meaning. Data abstraction hides irrelevant details; program-data insulation separates data definitions from application code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 5 | lines 17-25

> 1.2 Database System versus File System The traditional file processing system is file-directory structure supported by a conventional operating system. A file system organization of data lucks a number of major features of a database system, such as Data redundancy and inconsistency, Difficulty in accessing data, Data isolation, Integrity problems, Atomicity problems, Concurrent access anomalies. 1.3 Characteristics of the Database Approach The main characteristics of the database approach are Self-describing nature, Data abstraction and Insulation between programs and data, Support of multiple views of the data, sharing of data and multiuser transaction processing.

Fundamental of Database Systems module | PDF page 6 | lines 2-4

> Data abstraction and Insulation between programs and data- Data Abstraction is a process of hiding unwanted or irrelevant details from the end user. It provides a different view and helps in achieving data independence which is used to enhance the security of data.

Fundamental of Database Systems module | PDF page 8 | lines 8-15

2.4 Data Independence The three-schema architecture can be used to further explain the concept of data independence, which can be defined as the capacity to change the schema at one level of a database system without having to change the schema at the next higher level. The two types of data independence: 1. Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs. 2. Physical data independence is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas do not need be changed as well.

Question sources

Computer Science Exit Exam practice collection · Question 49

Question 211 | CS0050

Assume you are a database designer and you are given a task to specify entities, relationships, constraints on relations and data types for the data to be stored. At which level of the three-schema architecture
would this be addressed?

A. View level

B. Internal level

C. External level

D. Conceptual level

Hint

Trace which schema levels a proposed change affects.

Answer D: Conceptual level

The conceptual level describes the whole database and explicitly concentrates on entities, data types, relationships, operations and constraints while hiding physical storage. The stem repeats those conceptual responsibilities.

Why each choice is right or wrong

A - Incorrect: View level shows a user-group subset. (PDF pp. 7, 8.)

B - Incorrect: Internal level describes physical storage and access paths. (PDF pp. 7, 8.)

C - Incorrect: External is another name for user view, not the global description. (PDF pp. 7, 8.)

D - Correct: Conceptual is the level explicitly listing the requested structural components. (PDF pp. 7, 8.)

Primary note proof

Study-note section FDB04

The external level gives each user group a selected view. The conceptual level describes the whole logical database: entity types, relationships, data types and constraints. The

internal/physical level describes files, record placement and access paths. These are abstraction levels, distinct from client/server deployment tiers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 7 | lines 25-30

2.3 Three-schema Architecture and Data Independence The goal of the three-schema architecture is to separate the user applications from the physical database. In this architecture, schemas can be defined at the following three levels: 1. The internal level has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.

Fundamental of Database Systems module | PDF page 8 | lines 2-7

2. The conceptual level has a conceptual schema, which describes the structure of the whole database for users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. 3. The external or view level includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

Question sources

Computer Science Exit Exam practice collection · Question 50

Question 212 | NS0217

What is the purpose of normalization in database design?

A. To reduce redundancy and improve data integrity

B. To speed up query performance

C. To create complex relationships between tables

D. To encrypt sensitive data

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer A: Reduce avoidable redundancy and associated integrity anomalies.

FDB19 defines normalization as separating facts according to their determinants, reducing avoidable repetition and insertion, deletion and update anomalies. That supports both parts of A. It does not promise universally faster queries, deliberately create complexity for its own sake or encrypt content. Performance and security need their own workload and threat-model choices.

Why each choice is right or wrong

A - Correct: Normalization reduces repeated facts and the anomalies that threaten data consistency.

B - Incorrect: More joins may be needed, so faster query execution is not a universal normalization purpose.

C - Incorrect: Relationships follow fact dependencies; creating complexity is not the goal.

D - Incorrect: Normalization restructures facts and does not encrypt sensitive values.

Primary note proof

Study-note section FDB19

> Suppose Enrollment repeats StudentName for every course. An update anomaly requires changing every copy of the student's name; missed copies disagree. An insertion anomaly prevents recording a new student until some enrollment exists. A deletion anomaly loses the only record of a student's details when their last enrollment is deleted. Revoke anomaly is not a normalization anomaly.

Study-note section FDB19

> Normalization separates facts according to their determinants, reducing avoidable repetition and associated anomalies.

Study-note section FDB19

> Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design.

Question sources

Computer Science Exit Exam practice collection · Question 53

Question 213 | NS0295

A weak entity uses its partial key together with the key of an independently keyed owner. Which terms describe that associated entity set? Select all that apply.

A. Identifying set

B. Strong entity set

C. Neighbor set

D. Owner set

Hint

Check how a dependent entity obtains its complete identity.

Answer B, D: B. Strong entity set; D. Owner set

The weak entity lacks a complete independent identifier. Its owner's key combines with the partial key to identify it. Because the owner is independently keyed in this question, strong entity set and owner set both describe the associated set. Identifying describes the relationship connecting the sets, while neighbor does not state the necessary ownership and identification dependency.

Why each choice is right or wrong

A - Incorrect: Identifying is normally the relationship role rather than the defined name of the owner entity set.

B - Correct: The owner that supplies an independent identifying key is normally a strong entity set.

C - Incorrect: Neighbor does not express the necessary ownership and identification dependency.

D - Correct: Owner set is the associated entity set whose key identifies the weak entity together with its partial key.

Primary note proof

Study-note section FDB08

> A weak entity lacks a complete independent key. Its owner is normally a strong entity, so owner entity set and strong entity set may both describe the associated set. Its partial key, also called discriminator, distinguishes weak entities of the same owner. Identification combines the owner key and partial key.

Study-note section FDB08

> The weak entity participates totally in its identifying relationship.

Study-note section FDB08

> Chen notation uses a double rectangle, double identifying diamond and dashed underline for a partial key.

Question sources

Computer Science Exit Exam · 2018 · Question 12

Question 214 | NS0297

A language that allows the DBA or user to describe and name the entities, attributes, and relationships required for the application is_____

A. Transaction control language (TCL)

B. Data control language (DCL)

C. Data Manipulation Language (DML)

D. Data Definition Language (DDL)

Hint

Classify each SQL command by the kind of change it performs.

Answer D: Data Definition Language.

Describing and naming application entities, attributes and relationships defines schema structure. FDB26 assigns structural changes to DDL through CREATE, ALTER and DROP. TCL manages transaction boundaries, DCL controls privileges and DML retrieves or changes data. These categories have different effects, so DDL uniquely matches the schema-description role in the stem.

Why each choice is right or wrong

A - Incorrect: TCL manages commit, rollback and savepoints rather than defining schema elements.

B - Incorrect: DCL manages privileges through GRANT and REVOKE rather than describing entities and attributes.

C - Incorrect: DML retrieves or changes instance data rather than defining the application schema.

D - Correct: DDL defines and changes schema structure, including tables, attributes and constraints.

Primary note proof

Study-note section FDB26

> DDL changes structure: CREATE, ALTER, DROP, commonly RENAME and TRUNCATE in textbook classifications. DML retrieves or changes data: SELECT, INSERT, UPDATE, DELETE. Some texts separate SELECT as DQL. DCL manages privileges through GRANT and REVOKE. TCL includes COMMIT, ROLLBACK and SAVEPOINT. Classify a statement by its effect, not its length.

Question sources

Computer Science Exit Exam · 2018 · Question 14

Question 215 | NS0302

In relational-table terminology, which terms describe a unit of related field values treated together by an application? Select all that apply.

A. Rows

B. Field

C. Record

D. Database

Hint

Distinguish a relation's structure from its current contents and its descriptive metadata.

Answer A, C: A. Rows; C. Record

A tuple holds a set of related attribute values as one table unit. Row and record are synonymous names for that unit, so both alternatives qualify. A field is only an individual property, and a database is the larger coherent collection of related data. Grammatical number in Rows does not create a different relational concept from Record.

Why each choice is right or wrong

A - Correct: Rows contain related field values and are the relational counterpart of records.

B - Incorrect: A field is an individual property rather than a collection of related fields.

C - Correct: Record is the stated unit that groups related fields for application use.

D - Incorrect: A database contains a larger coherent collection of data rather than just one record.

Primary note proof

Study-note section FDB03

> A relation is a set of tuples, represented as a table. A tuple/row/record contains related attribute values; an attribute/column/field describes a property. Degree counts columns; cardinality counts rows. Classical relations have distinct tuples, atomic cells and domains for attributes; row order has no meaning. Column order has no logical meaning when attributes are identified by name. SQL tables can contain duplicate rows unless constrained.

Question sources

Computer Science Exit Exam · 2018 · Question 20

Question 216 | NS0315

Which template creates a table with one column? Treat table_name, column_name and column_type as placeholders for valid identifiers and a data type.

A. CREATE table_name (column_name, column type);

B. CREATE table_name (column_type, column name);

C. CREATE TABLE table_name (column_name column_type);

D. CREATE TABLE table_name (column_type, column name);

Hint

Distinguish table definition changes from changes to stored rows.

Answer C: C. CREATE TABLE table_name (column_name column_type);

CREATE TABLE introduces a new table definition. Within its parentheses, each column declaration places the column name before its data type, without a comma between those two parts. Commas separate whole declarations. Option C matches this pattern. A and B omit TABLE, while D reverses and separates the intended name/type pair and therefore fails the requested template.

Why each choice is right or wrong

A - Incorrect: TABLE is absent and the column declaration is split incorrectly.

B - Incorrect: TABLE is absent and the name/type order is reversed.

C - Correct: The name/type pair forms one complete column declaration.

D - Incorrect: The type/name order and comma do not form the intended declaration.

Primary note proof

Study-note section FDB30

> CREATE TABLE Employee (EmpID INTEGER PRIMARY KEY, Name VARCHAR(80)); declares two columns. The name and its type stay together; the comma separates whole column declarations. To add a primary key later: ALTER TABLE Employee ADD PRIMARY KEY (EmpID); existing data must be unique and non-NULL. A table cannot gain a second primary key without replacing the existing constraint.

Question sources

Computer Science Exit Exam · 2018 · Question 40

Question 217 | NS0320

From the following alternatives which one is FALSE about database system?

A. It is very difficult to protect a file under the system

B. In database system the user is not required to write the procedures

C. In database system, DBMS provides a good protection mechanism.

D. It contains a wide variety of sophisticated techniques to store and retrieve the data.

Hint

Review the database approach and match each claimed benefit or cost to its mechanism.

Answer A: The claim that protection is very difficult under a database system is false.

FDB01 contrasts unmanaged file difficulties with DBMS authorization, shared integrity, transactions and recovery. A therefore reverses a stated database benefit. B is understood in the data-access sense: declarative requests need not prescribe retrieval procedures, as FDB36 explains; it does not mean applications never contain code. Protection and sophisticated storage/access facilities in C and D are consistent with the note.

Why each choice is right or wrong

A - Correct: This reverses the note's benefit: a DBMS provides coordinated protection rather than inherently making protection very difficult.

B - Incorrect: Declarative data requests need not prescribe retrieval procedures; this does not exclude ordinary application code.

C - Incorrect: Authorization and controlled data access are explicit DBMS protection mechanisms.

D - Incorrect: DBMS storage and access facilities include indexes, buffers, query processing and other sophisticated mechanisms.

Primary note proof

Study-note section FDB01

> A database is a coherent collection of related facts with meaning for an application. A DBMS is software that defines, stores, retrieves, updates, protects and shares that data. The database is the content; the DBMS manages it. A database system also includes applications, people and infrastructure.

Study-note section FDB01

> Traditional file applications define their own formats and access procedures.

Study-note section FDB01

> A DBMS centralizes metadata and supports multiple views, authorization, transactions, backup and recovery.

Related reading (context, not primary note proof)

Study-note section FDB36

Study-note section FDB25

Question sources

Computer Science Exit Exam · 2018 · Question 45

Question 218 | NS0329

Consider a directed line(->) from the relationship set advisor to both entity sets instructor and student. This indicates _____ cardinality

A. Many to one

B. Many to many

C. One to one

D. One to many

Hint

Read the relationship in both directions and check participation constraints.

Answer C: One-to-one under the common ER arrow convention.

FDB09 explicitly discusses arrows from advisor to both Instructor and Student. Under its common convention, each arrow imposes a maximum of one on the corresponding end. Both ends are therefore maximum-one, producing 1:1. This does not imply total participation, and another diagram notation would require a legend, but the complete note supplies the convention for this example.

Why each choice is right or wrong

A - Incorrect: Many-to-one permits many at one end, contrary to maximum-one arrows at both ends.

B - Incorrect: Many-to-many imposes neither of the maximum-one constraints stated by the arrows.

C - Correct: Maximum one at both ends produces a one-to-one relationship.

D - Incorrect: One-to-many permits multiple associations at one end rather than maximum one at both.

Primary note proof

Study-note section FDB09

> Read business rules in both directions. If each employee belongs to one department and each department has many employees, Employee→Department is N:1; Department→Employee is 1:N. Building→Location could also be N:1, so names alone cannot select a unique example. Under the common arrow convention, arrows from advisor to both Instructor and Student impose maximum-one at both ends, giving 1:1. Without a notation convention, arrows are unsafe evidence.

Question sources

Computer Science Exit Exam · 2018 · Question 56

Question 219 | NS0339

What is the basic goal of Normalization?

A. To decrease data integrity

B. To reduce redundancy

C. To maximize transitive dependency

D. To increase dependency

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer B: Reduce avoidable redundancy.

Normalization separates facts by their determinants to reduce repeated storage and the anomalies that repetition can cause. FDB19 therefore supports reducing redundancy as the basic listed goal. Decreasing integrity is the opposite intended effect, and maximizing transitive or unspecified dependencies is not a normalization objective. Proper dependencies are analyzed rather than increased indiscriminately.

Why each choice is right or wrong

A - Incorrect: Normalization aims to improve integrity by reducing anomalies, not decrease it.

B - Correct: Reducing avoidable repeated facts is a basic normalization goal.

C - Incorrect: Problematic nonkey transitive dependencies are removed or separated rather than maximized.

D - Incorrect: Dependencies are analyzed to place facts correctly; increasing them indiscriminately is not the objective.

Primary note proof

Study-note section FDB19

> Suppose Enrollment repeats StudentName for every course. An update anomaly requires changing every copy of the student's name; missed copies disagree. An insertion anomaly prevents recording a new student until some enrollment exists. A deletion anomaly loses the only record of a student's details when their last enrollment is deleted. Revoke anomaly is not a normalization anomaly.

Study-note section FDB19

> Normalization separates facts according to their determinants, reducing avoidable repetition and associated anomalies.

Study-note section FDB21

> Remove dependencies of nonprime facts on nonkey determinants.

Question sources

Computer Science Exit Exam · 2018 · Question 70

Question 220 | NS0344

For a relation with ID, CITY and NAME, ID is guaranteed unique and non-NULL while CITY and NAME may repeat. Which listed sets are superkeys? Select all that apply.

A. NAME

B. ID

C. CITY

D. CITY, ID

Hint

Test uniqueness and minimality separately when classifying keys.

Answer B, D: B. ID; D. CITY, ID

A superkey is any set of attributes guaranteed to identify a tuple. ID has that guarantee, and the larger set containing CITY and ID keeps it. Both sets therefore qualify, even though the larger one is not minimal. CITY and NAME may repeat and cannot independently identify all tuples. Minimality would distinguish candidate keys, but it is not required for superkeys.

Why each choice is right or wrong

A - Incorrect: Repeated names are permitted, so NAME is not guaranteed unique.

B - Correct: ID uniquely identifies every tuple.

C - Incorrect: Repeated cities are permitted, so CITY is not guaranteed unique.

D - Correct: Including unique ID preserves identification even with the extra CITY attribute.

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB07

> If ID is unique, both {ID} and {ID,City} are superkeys, but only {ID} is minimal.

Study-note section FDB07

> A table can have several candidates and foreign keys but at most one declared primary-key constraint, possibly containing several columns.

Question sources

Computer Science Exit Exam · 2018 · Question 77

Question 221 | NS0357

_____is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

A. Super key

B. Candidate key

C. Social security key

D. Primary key

Hint

Test uniqueness and minimality separately when classifying keys.

Answer D: Primary key.

FDB07 defines a primary key as the selected candidate key, with the remaining candidates called alternate keys. The stem asks specifically for the designer's selection rather than every unique attribute set. Superkey is broader and candidate key denotes eligibility before selection. A social-security identifier could be chosen, but is not the generic name of that role.

Why each choice is right or wrong

A - Incorrect: A superkey is any unique identifier set and may include unnecessary attributes.

B - Incorrect: Candidate keys are eligible minimal identifiers, not specifically the one selected for the primary role.

C - Incorrect: A social-security identifier is a possible attribute choice, not the generic selected-key category.

D - Correct: The selected candidate key is the primary key.

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Question sources

Computer Science Exit Exam · 2018 · Question 95

Question 222 | NS0627

What is the difference between a primary key and a foreign key?

Hint

Test uniqueness and minimality separately when classifying keys.

Answer: A primary key is the selected unique non-NULL identifier; a foreign key references an eligible key.

FDB07 defines the primary key as a selected candidate key. FDB15 requires non-NULL foreign-key values to match an eligible referenced unique/candidate key; foreign keys need not be unique and may permit NULL.

Primary note proof

Study-note section FDB07

> A superkey uniquely identifies a tuple. A candidate key is a minimal superkey: removing any component loses uniqueness. Minimal means no unnecessary attribute, not necessarily the fewest columns among all keys. A primary key is one selected candidate key; remaining candidates are alternate keys. A composite key contains multiple attributes and need not be the chosen primary key.

Study-note section FDB15

> Referential integrity requires a non-NULL foreign-key value to match an eligible referenced unique/candidate key; self-references are allowed.

Study-note section FDB15

> Entity integrity requires every primary-key component to be non-NULL.

Related reading (context, not primary note proof)

Study-note section FDB16

Question sources

Database Fundamentals review questions · Question 1

Question 223 | NS0628

Why is database normalization important? What problems does it solve?

Hint

Distinguish dependency-based design problems from the contents of one sample table.

Answer: Normalization reduces repeated facts and insertion, deletion and update anomalies.

Repeating a student name across enrollments risks inconsistent updates, prevents storing some facts without unrelated rows and can lose a fact when its last repeated row is deleted. Normalization separates facts by their determinants to reduce these anomalies.

Primary note proof

Study-note section FDB19

> Suppose Enrollment repeats StudentName for every course. An update anomaly requires changing every copy of the student's name; missed copies disagree. An insertion anomaly prevents recording a new student until some enrollment exists. A deletion anomaly loses the only record of a student's details when their last enrollment is deleted. Revoke anomaly is not a normalization anomaly.

Study-note section FDB19

> Normalization separates facts according to their determinants, reducing avoidable repetition and associated anomalies.

Study-note section FDB19

> Adding a generated row ID may make insertion technically possible, but does not remove repeated facts or fix the dependency design.

Question sources

Database Fundamentals review questions · Question 2

Question 224 | NS0629

What happens if you UPDATE a row but forget to include a WHERE clause?

Hint

Track which rows each data-changing command affects.

Answer: UPDATE without WHERE targets every row, subject to constraints and transaction outcome.

FDB29 states that an UPDATE without WHERE targets every row. The SET expressions are therefore applied to all rows rather than one intended row; constraints or errors can still prevent completion.

Primary note proof

Study-note section FDB29

> INSERT INTO Hotel(HotelID,HotelName,SubCity) VALUES (1,'Hilton','Yeka'); pairs each named column with a compatible value. Counts and types must match; keys and other constraints still apply. Without a column list, values must follow the table's declared column order.

Study-note section FDB29

> UPDATE without WHERE targets every row.

Study-note section FDB29

> UPDATE Book SET Price = 600 WHERE ID = 'B001'; changes matching rows.

Related reading (context, not primary note proof)

Study-note section FDB16

Question sources

Database Fundamentals review questions · Question 3

Question 225 | NS0630

Why would you use a ROLLBACK command in a transaction?

Hint

Match each transaction guarantee to its precise failure or interference scenario.

Answer: ROLLBACK undoes uncommitted transaction work, for example after a failure or unwanted change.

FDB37 contrasts COMMIT, which completes transaction work, with ROLLBACK, which undoes uncommitted work. Rollback restores the prior transactional state after errors or a decision not to keep the changes; it does not erase other users' committed transactions.

Primary note proof

Study-note section FDB37

> For a transfer, subtracting 100 from one balance and adding 100 to another must commit together; losing either half violates atomicity and may violate the total-balance invariant. BEGIN starts work, COMMIT completes it, ROLLBACK undoes uncommitted work and SAVEPOINT marks an intermediate rollback point. A failed transaction may roll back while other users' committed transactions remain. SQL transaction syntax and error-handling constructs differ by DBMS; @@ERROR/IF is SQL Server-specific.

Question sources

Database Fundamentals review questions · Question 4

Question 226 | NS0631

What is the difference between DELETE and DROP?

Hint

Classify each SQL command by the kind of change it performs.

Answer: DELETE removes rows and retains the table; DROP TABLE removes the table object.

FDB26 distinguishes data deletion from structural removal. DELETE may target selected rows with WHERE, or all rows without it, while leaving the table defined. DROP TABLE removes that relation object.

Primary note proof

Study-note section FDB26

> DELETE removes rows and keeps the table; DROP TABLE removes the table object; ALTER changes its definition. TRUNCATE empties a table using implementation-specific mechanisms. PostgreSQL allows transactional rollback of TRUNCATE, so no rollback information is retained is not a universal SQL property. SQL Server permits DROP DATABASE Sales, suppliers; that comma-list syntax is vendor-specific. SQLite's .databases lists attached names/files in its command-line shell; .database is not the documented command.

Study-note section FDB29

> UPDATE without WHERE targets every row.

Study-note section FDB29

> DELETE FROM r; removes every row but leaves r defined.

Question sources

Database Fundamentals review questions · Question 5

Advanced Database Systems

Questions & Worked Answers

239 questions + 4 source variants

Question 1 | Q00005

_____ is the process of selecting appropriate query execution strategy?

A. Query graph

B. Query optimization

C. Query tree

D. Query processing

Hint

Separate query representation, strategy selection and execution.

Answer B: Query optimization

Selecting an efficient execution strategy is optimization. A tree or graph represents the query; processing is the broader transformation and execution activity.

Why each choice is right or wrong

A - Incorrect: A graph corresponds to relational calculus, not strategy selection. (PDF pp. 27, 34.)

B - Correct: The module explicitly defines optimization as choosing an efficient strategy. (PDF pp. 27, 34.)

C - Incorrect: A query tree represents relational algebra and operations. (PDF pp. 27, 34.)

D - Incorrect: Processing includes decomposition and execution as well as optimization. (PDF pp. 27, 34.)

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB17

> Query processing is the whole activity; selecting a suitable efficient strategy is optimization.

Study-note section ADB-S01

A query graph represents the predicate relationships of a relational-calculus expression without itself prescribing an operator execution order.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 27 | lines 7-13

What is Query Optimization?
✓ The activity of choosing a single —efficient‖ execution strategy (from hundreds) as determined by database catalog statistics.
✓ The aim is to answer which relational algebra expression, equivalent to the given query, will lead to the most efficient solution plan?
✓ For each algebraic operator, what algorithm (of several available) do we use to compute that operator?

Fundamental of Database Systems module | PDF page 34 | lines 5-7

2. Apply heuristics rules to optimize the internal representation.
3. A query execution plan is generated to execute groups of operations based on the access paths available on the files involved in the query.

Fundamental of Database Systems module | PDF page 34 | lines 17-27

• Query tree:
1. A tree data structure that corresponds to a relational algebra expression. It represents the input relations of the query as leaf nodes of the tree, and represents the relational algebra operations as internal nodes.
• An execution of the query tree consists of executing an internal node operation whenever its operands are available and then replacing that internal node by the relation that results from executing the operation.
• Query graph:
1. A graph data structure that corresponds to a relational calculus expression. It does not indicate an order on which operations to perform first. There is only a single graph corresponding to each query.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 5

Question 2 | Q00022

Which update-timing policy permits a transaction's changes to reach the persistent database before the transaction commits?

A. In-Place Update

B. Immediate update

C. Shadow update

D. Deferred update

Hint

Separate update timing from update location and follow the commit boundary.

Answer B: Immediate update

Immediate update permits a transaction's changes to be written to the persistent database before commit. If that transaction later aborts, recovery may need to undo those changes. Deferred update keeps uncommitted changes outside the persistent database. In-place update describes overwriting an existing location, while shadow update describes writing a separate version. Those location strategies do not, by their names alone, specify the timing policy requested here. Immediate update need not flush every cache write immediately.

Why each choice is right or wrong

A - Incorrect: Describes overwriting the existing location, not the commit-timing policy.

B - Correct: Correct: immediate update permits persistent database changes before commit and may require undo.

C - Incorrect: Describes writing a separate version while preserving the committed one, not the requested timing policy.

D - Incorrect: Deferred update excludes uncommitted changes from the persistent database.

Primary note proof

Study-note section ADB29

> Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO. Redo-only and NO-UNDO/REDO mean the same thing, so those cannot be separate unique answers. Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Study-note section ADB29

> Shadow/copy-on-write update writes a new location while preserving the committed version.

Study-note section ADB29

> In-place update overwrites the original location.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 110 | lines 14-30

> ◦ Deferred Update:
> • All transaction updates are recorded in the local workspace (cache)
> • All modified data items in the cache is then written after a transaction ends its execution or after a fixed number of transactions have completed their execution.
> • During commit the updates are first recorded on the log and then on the database
> • If a transaction fails before reaching its commit point undo is not needed

because it didn‘t change the database yet
• If a transaction fails after commit (writing on the log) but before finishing saving to the data base redoing is needed from the log
◦ Immediate Update:
• As soon as a data item is modified in cache, the disk copy is updated.
• These update are first recorded on the log and on the disk by force writing, before the database is updated
• If a transaction fails after recording some change to the database but before reaching its commit point , this will be rolled back

Fundamental of Database Systems module | PDF page 111 | lines 2-11

◦ Shadow update:
• The modified version of a data item does not overwrite its disk copy but is written at a separate disk location.
• Multiple version of the same data item can be maintained
• Thus the old value (before image BFIM) and the new value (AFIM) are kept in the disk
• No need of Log for recovery
◦ In-place update: The disk version of the data item is overwritten by the cache version.
• Information for ARIES to accomplish recovery procedures includes:

Question sources

MOE Computer Science Exit Exam · 2016 · Question 22

Question 3 | Q00026

Which query-optimization approach applies relational-algebra properties and equivalences to rewrite a query and reduce intermediate work?

A. Database storage without query rewriting

B. Heuristic approach

C. Deterministic execution without query rewriting

D. Physical-plan cost estimation

Hint

Consider which equivalent transformation changes intermediate work and what assumptions preserve meaning.

Answer B: Heuristic approach

Heuristic optimization uses algebraic rules to reduce intermediate work, for example applying selective filters earlier, replacing a product plus join predicate with a join and projecting away unneeded columns. Storage and execution without rewriting do not perform those transformations. Cost estimation predicts resources for candidate plans rather than naming the algebraic rewrite approach itself. A practical optimizer can combine rewriting and costing, but the question isolates the rule-based transformation role.

Why each choice is right or wrong

A - Incorrect: Incorrect: storage without rewriting does not transform the algebraic query representation.

B - Correct: Correct: heuristic rewriting uses operator properties and equivalences to reduce intermediate work.

C - Incorrect: Incorrect: repeatable execution without rewriting does not apply the requested query transformations.

D - Incorrect: Incorrect: cost estimation predicts candidate-plan resource use rather than identifying the algebraic rewrite approach.

Primary note proof

Study-note section ADB18

> Heuristics reduce intermediate work: push selective row filters toward base tables; replace product followed by a join predicate with a join; project unneeded columns away while retaining join/filter columns; reorder inner joins when equivalence permits; reuse common expressions when worthwhile. Split conjunctive selections when each predicate can be applied earlier to its own input.

Study-note section ADB19

> A cost model estimates resources for a candidate plan: page I/O, CPU, memory, temporary storage and network traffic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 31 | lines 19-23

> • Heuristics Approach uses the knowledge of the characteristics of the relational algebra operations and the relationship between the operators to optimize the query.
> • Thus the heuristic approach of optimization will make use of:
> ◦ Properties of individual operators
> ◦ Association between operators

Question sources

MOE Computer Science Exit Exam · 2016 · Question 26

Question 4 | Q00046

In which phase of the two phase locking protocol can a transaction obtain locks?

A. Release phase

B. Roll back phase

C. Shrinking phase

D. Growing phase

Hint

Track the first lock release and the transaction-completion boundary.

Answer D: Growing phase

Two-phase locking acquires locks in the growing phase, then releases them in the shrinking phase without acquiring new ones.

Why each choice is right or wrong

A - Incorrect: Release is the unlocking portion, not acquisition. (PDF pp. 86.)

B - Incorrect: Rollback is not one of the two acquisition/release phases. (PDF pp. 86.)

C - Incorrect: Shrinking forbids new locks. (PDF pp. 86.)

D - Correct: Growing allows acquisition and forbids release. (PDF pp. 86.)

Primary note proof

Study-note section ADB12

> Basic 2PL has a growing phase that acquires/upgrades locks and a shrinking phase that releases/downgrades them. Once any lock is released, no new lock may be acquired. The lock point is the instant the final lock is obtained. 2PL guarantees conflict serializability for well-formed transactions, but can deadlock and basic 2PL alone need not prevent cascading aborts.

Study-note section ADB12

> Two-phase locking orders access to data; two-phase commit coordinates a distributed commit decision.

Study-note section ADB12

> Strict 2PL retains exclusive locks until commit/abort.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 86 | lines 16-24

> ◦ Locking (Growing) Phase:
> • A transaction applies locks (read or write) on desired data items one at a time.

• acquires all locks but cannot release any locks.
◦ Unlocking (Shrinking) Phase:
• A transaction unlocks its locked data items one at a time.
• Releases locks but cannot acquire any new locks.
● Requirement:
◦ For a transaction these two phases must be mutually exclusively, that is, during

Question sources

MOE Computer Science Exit Exam · 2016 · Question 46

Computer Science Exit Exam practice questions · Question 38

Question 5 | Q00053

Which one of the following is not problem if concurrency is not controlled properly?

A. Lost update problem

B. Consistent database

C. Incorrect summary

D. Dirty read

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer B: Consistent database

Lost update, temporary update/dirty read, and incorrect summary are concurrency problems. A consistent database is the desired outcome of concurrency control.

Why each choice is right or wrong

A - Incorrect: Lost update is explicitly a concurrency anomaly. (PDF pp. 68, 70, 79.)

B - Correct: Consistency is a desired state, so this is the exception. (PDF pp. 68, 70, 79.)

C - Incorrect: Incorrect summary mixes values from different update stages. (PDF pp. 68, 70, 79.)

D - Incorrect: Dirty read exposes an update before its transaction has safely committed. (PDF pp. 68, 70, 79.)

Primary note proof

Study-note section ADB08

Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Study-note section ADB06

Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure.

Study-note section ADB06

A transaction is a logical unit of database work, including read-only work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 68 | lines 4-7

i. The Lost Update Problem
◦ This occurs when two transactions that access the same database items have their operations interleaved in a way that makes the value of some database item incorrect.

Fundamental of Database Systems module | PDF page 70 | lines 2-12

ii. The Temporary Update (or Dirty Read) Problem
◦ This occurs when one transaction updates a database item and then the transaction fails for some reason .
◦ The updated item is accessed by another transaction before it is changed back to its original value. Based on the above scenario:
iii. The Incorrect Summary Problem
◦ If one transaction is calculating an aggregate summary function on a number of records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.
Example: T1 would like to add the values of A=10, B=20 and C=30. after the values are read by

Fundamental of Database Systems module | PDF page 79 | lines 4-15

● Transaction Processor is divided into:
◦ A concurrency-control manager, or scheduler, responsible for assuring isolation of transactions
◦ A logging and recovery manager, responsible for the durability of transactions.
● The scheduler (concurrency-control manager) must assure that the individual actions of multiple transactions are executed in such an order that the net effect is the same as if the transactions had in fact executed one-at-a-time.
1. Purpose of Concurrency Control
◦ To enforce Isolation (through mutual exclusion) among conflicting transactions.
◦ To preserve database consistency through consistency preserving execution of transactions.
◦ To resolve read-write and write-write conflicts.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 53

Question 6 | Q00078

Which one of the following is not a condition for two operations in a schedule to be said to conflict?

A. If they access the same item X

B. If they belong to the same transactions

C. If Write_item (X) is at least one of the operations

D. If they belong to different transactions

Hint

Distinguish transaction ordering, conflict conditions and full execution equivalence.

Answer B: If they belong to the same transactions

Conflict requires different transactions, the same item, and at least one write. Membership in the same transaction is therefore the exception.

Why each choice is right or wrong

A - Incorrect: Accessing the same item is required. (PDF pp. 70, 71.)

B - Correct: Same-transaction operations fail the different-transaction requirement. (PDF pp. 70, 71.)

C - Incorrect: At least one write is required. (PDF pp. 70, 71.)

D - Incorrect: Different transactions is required, so D is not the exception. (PDF pp. 70, 71.)

Primary note proof

Study-note section ADB09

> A serial schedule executes each transaction's operations consecutively. A serializable schedule is equivalent to some serial order; it may interleave transactions. Equality of one final value is insufficient evidence because different executions can accidentally coincide. Two operations conflict when they belong to different transactions, access the same item and at least one writes. Read/read pairs do not conflict.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 70 | lines 2-12

> ii. The Temporary Update (or Dirty Read) Problem
> ◦ This occurs when one transaction updates a database item and then the transaction fails for some reason .
> ◦ The updated item is accessed by another transaction before it is changed back to its original value. Based on the above scenario:
> iii. The Incorrect Summary Problem
> ◦ If one transaction is calculating an aggregate summary function on a number of

records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.
Example: T1 would like to add the values of A=10, B=20 and C=30. after the values are read by

Fundamental of Database Systems module | PDF page 71 | lines 15-26

- Recoverable schedule:
  - One where no committed transaction needs to be rolled back.
  - A schedule S is recoverable if no transaction Tj in S commits until all transactions Ti that have written an item that Tj reads have committed. Examples,
- Sc: r1(X); w1(X); r2(X); r1(Y);w2(x);c2;a1; not recoverable
- Sd: r1(X); w1(X); r2(X); r1(Y); w2(X);w1(Y); c1; c2;
- Se: r1(X); w1(X); r2(X); r1(Y); w2(x) ; w1(Y); a1; a2;
- Cascadeless schedule:
  - One where every transaction reads only the items that are written by committed transactions. Eg.
- Sf: r1(X); w1(X); r1(Y); c1; r2(X); w2(X);w1(Y); c2;
- Strict Schedules:

Question sources

MOE Computer Science Exit Exam · 2016 · Question 78

Question 7 | Q00079

During restart recovery after a crash, which operation removes the database effects of a transaction that had not committed?

A. View

B. Commit

C. Rollback

D. Redo

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer C: Rollback

Atomicity requires removing the effects of an incomplete transaction after a crash. Rollback performs that undo work, restoring before-images as needed. Commit instead accepts successful transaction effects, and a view is a query representation rather than a recovery action. Redo installs logged after-images to recover missing work; it does not name the removal of uncommitted effects requested here. These actions occur during restart recovery after the system is operating again.

Why each choice is right or wrong

A - Incorrect: Incorrect: a view is not an operation that reverses transaction effects.

B - Incorrect: Incorrect: commit accepts transaction effects rather than undoing the incomplete work.

C - Correct: Correct: rollback undoes the effects of the uncommitted transaction.

D - Incorrect: Incorrect: redo installs logged after-images; the requested removal requires undo/rollback.

Primary note proof

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects. Rollback handles failed/uncommitted work; redo preserves committed work missing from pages; checkpoints reduce log-search work; backups provide another stored copy.

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward.

Study-note section ADB25

> A view is a stored query definition, not encryption.

Related reading (context, not primary note proof)

Study-note section ADB06

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

> • Recovery from catastrophic
> • If there is extensive damage to the wide portion of the database
> • This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
> • Recovery from non-catastrophic failure
> • When the database is not physically damaged but has be come in consistent
> • The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
> • For instance,
> • If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
> • If transaction had not committed at failure time, recovery manager has to undo

(rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 107 | lines 16-28

• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:
• A set of transactions records their updates in the log.
• At commit point under WAL scheme these updates are saved on database disk.
• No undo is required because no AFIM is flushed to the disk before a transaction commits.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 79

Question 8 | Q00081

A system is in a _____ state if there exists a set of transactions such that every transaction in the set is waiting for another transaction in the set.

A. Idle

B. Waiting

C. Deadlock

D. Ready

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer C: Deadlock

Mutual waiting among transactions for locks held by one another is a deadlock condition.

Why each choice is right or wrong

A - Incorrect: Idle does not express mutual lock waiting. (PDF pp. 87.)

B - Incorrect: Waiting alone need not involve a cycle. (PDF pp. 87.)

C - Correct: Deadlock is mutual waiting that prevents progress. (PDF pp. 87.)

D - Incorrect: Ready does not describe transactions blocked on one another. (PDF pp. 87.)

Primary note proof

Study-note section ADB13

> Deadlock requires a waiting cycle: T1 holds X and wants Y; T2 holds Y and wants X. In a wait-for graph, an edge Ti→Tj means Ti waits for a lock held by Tj. A cycle indicates deadlock under the ordinary lock-wait model. Abort a suitable victim, undo its effects and release its locks; repeat if other cycles remain. Aborting every deadlocked transaction also breaks cycles but wastes more work.

Study-note section ADB13

> Fair queues, sensible victim selection and retaining original transaction timestamps on retry reduce starvation; deadlock freedom alone does not prove fairness.

Study-note section ADB13

> A timeout aborts long waits but may mistake slow progress for deadlock.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 87 | lines 2-6

> Dealing with Deadlock and Starvation
> - Deadlock
>   - It is a state that may result when two or more transaction are each waiting for locks held by the other to be released
>   - Example :

Question sources

MOE Computer Science Exit Exam · 2016 · Question 81

Question 9 | Q00113

Users may request permitted aggregates but must not infer protected individual values by subtracting overlapping aggregate answers. Which security measure specifically addresses this indirect-disclosure threat?

A. Inference control

B. Access control

C. Integrity control

D. Encryption

E. None of the above

Hint

Identify whether the scenario concerns identity, permission, disclosure or valid values.

Answer A: A. Inference control

Inference control targets learning protected individual facts from aggregate answers. For example, subtracting a sum excluding one person from a sum including that person can reveal their value even when their row is hidden. Ordinary access control grants or denies actions, integrity validates data and encryption protects encoded content. The specified indirect-disclosure threat identifies inference control specifically.

Why each choice is right or wrong

A - Correct: This directly prevents disclosure of individual facts inferred from allowed aggregates.

B - Incorrect: Authorization alone does not prevent inference from otherwise allowed answers.

C - Incorrect: Integrity constrains valid data rather than indirect disclosure.

D - Incorrect: Encryption does not stop an authorized user combining decrypted aggregate answers.

E - Incorrect: Inference control directly matches the threat.

Primary note proof

Study-note section ADB24

> Authentication establishes identity; authorization/access control determines permitted actions. Encryption protects stored or transmitted content; integrity constraints restrict valid data; auditing records actions. A user who can view but cannot change a record has restricted access privileges. A view can restrict visible rows/columns, but its privilege and execution rules must support that restriction.

Study-note section ADB24

> Inference control concerns learning protected individual facts indirectly from allowed aggregate answers.

Study-note section ADB24

> Releasing SUM for a group and SUM for the same group excluding one known person may disclose that person's value by subtraction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 52 | lines 22-26

> 2.2 Database Security
> - Views or restricted subschemas
> - Authorization rules to identify users and the actions they can perform
> - User-defined procedures (and rule systems) to define additional constraints or limitations in using the database

Fundamental of Database Systems module | PDF page 53 | lines 4-9

- Database security is about controlling access to information
- Some information should be available freely
- Other information should only be available to certain people or groups
- Many aspects to consider for security
- Legal issues
- Physical security

Question sources

Addis Ababa University mock exam · Question 15

Question 10 | Q00114

Which approach rewrites a query's internal representation using relational-algebra equivalences, without relying on integrity-constraint deductions or comparison of estimated plan costs?

A. Systematic plan-cost estimation

B. Semantic query optimization

C. Cost Estimation

D. Heuristic rules

E. Both A and C

Hint

Consider which equivalent transformation changes intermediate work and what assumptions preserve meaning.

Answer D: Heuristic rules

Heuristic rules rewrite algebraic representations using equivalences, such as applying suitable selections early or replacing a product followed by a join condition with a join. Semantic optimization additionally reasons from trusted constraints. Cost estimation predicts resources for candidate plans, and doing that systematically remains a costing activity. The stem isolates equivalence-based rewriting without those other bases, so D is correct and combining the two cost-estimation descriptions does not change the answer.

Why each choice is right or wrong

A - Incorrect: Incorrect: systematic cost estimation predicts plan costs rather than performing the specified equivalence-based rewriting.

B - Incorrect: Incorrect: semantic optimization additionally uses trusted constraints, which the stem excludes.

C - Incorrect: Incorrect: cost estimation compares resource predictions for candidate plans.

D - Correct: Correct: heuristic rules apply the stated relational-algebra equivalences.

E - Incorrect: Incorrect: A and C both describe costing, not the isolated algebraic-rewrite approach.

Primary note proof

Study-note section ADB18

> Heuristics reduce intermediate work: push selective row filters toward base tables; replace product followed by a join predicate with a join; project unneeded columns away while retaining join/filter columns; reorder inner joins when equivalence permits; reuse common expressions when worthwhile. Split conjunctive selections when each predicate can be applied earlier to its own input.

Study-note section ADB18

> Semantic optimization additionally uses trusted constraints, such as a key proving a join cannot multiply matching rows. Cost-based optimization estimates candidate plans instead of relying on rules alone.

Study-note section ADB19

> A cost model estimates resources for a candidate plan: page I/O, CPU, memory, temporary storage and network traffic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 34 | lines 5-7

> 2. Apply heuristics rules to optimize the internal representation.
> 3. A query execution plan is generated to execute groups of operations based on the access paths available on the files involved in the query.

Fundamental of Database Systems module | PDF page 42 | lines 19-22

> ◦ Estimate and compare the costs of executing a query using different execution strategies and choose the strategy with the lowest cost estimate. (Compare to heuristic query optimization)
> • Issues

Question sources

Addis Ababa University mock exam · Question 16

Question 11 | Q00115

Which of the following is not true about locks?

A. Locks with large granularity are easier for the DBMS to administer.

B. Locks with small granularity cause more conflicts.

C. Locks with large granularity produce fewer details for the DBMS to track.

D. Locks may have a table-level granularity.

E. Locks may have a database-level granularity.

Hint

Consider conflicts, private work and validation costs, or lock-management overhead.

Answer B: Locks with small granularity cause more conflicts.

Fine granularity raises concurrency, whereas coarse granularity locks more data with fewer lock entries. Thus the claim that smaller units cause more conflicts reverses the tradeoff.

Why each choice is right or wrong

A - Incorrect: A single coarse lock can replace many fine locks and reduce management work. (PDF pp. 83, 95.)

B - Correct: Fine units allow more independent access, so this is the false claim. (PDF pp. 83, 95.)

C - Incorrect: Coarse locks reduce the number of entries tracked. (PDF pp. 83, 95.)

D - Incorrect: The file/relation granularity in the hierarchy permits table-level locking. (PDF pp. 83, 95.)

E - Incorrect: Entire-database locking is expressly listed. (PDF pp. 83, 95.)

Primary note proof

Study-note section ADB16

> Fine-grained row locks permit more independent operations but require tracking many locks. Coarse table/database locks simplify management but block unrelated rows and reduce concurrency. Intention locks advertise planned descendant locks in a hierarchy; escalation replaces many small locks with a coarser one. Small granularity does not inherently cause more logical conflicts. Choose granularity from contention and management overhead, not a universal rule.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 83 | lines 2-8

> • Lock Manager:
> • Managing locks on data
> items.
> • Lock table:
> • Lock manager uses it to store the identify of transaction locking a data
> item, the data item, lock mode and pointer to the next data item locked.
> One simple way to implement a lock table is through linked list.

Fundamental of Database Systems module | PDF page 95 | lines 14-28

> • A lockable unit of data defines its granularity
> • Granularity can be coarse (entire database) or it can be fine (an attribute of a relation).
> • Example of data item granularity:
> • A field of a database record
> • A database record
> • A disk block/ page

• An entire file
• The entire database
• Data item granularity significantly affects concurrency control performance.
• Thus, the degree of concurrency is low for coarse granularity and high for fine granularity.
• Example:
• A transaction that expects to access most of the pages in a file should probably set a lock on the entire file , rather than locking individual pages or records
• If a transaction that requires to access relatively few pages of the file , it is better

Question sources

Addis Ababa University mock exam · Question 17

Question 12 | Q00116

For every transaction T participating in the schedule, if all the operations of T are executed consecutively in the schedule. The schedule is

A. Cascaded schedule

B. Serial schedule

C. Cascadeless schedule

D. Recoverable schedule

E. None

Hint

Distinguish transaction ordering, conflict conditions and full execution equivalence.

Answer B: Serial schedule

Consecutive execution of every transaction's operations is the definition of a serial schedule. Recoverability and cascadelessness instead constrain dependencies on commits.

Why each choice is right or wrong

A - Incorrect: Cascading concerns dependent rollbacks, not contiguous execution. (PDF pp. 72, 71.)

B - Correct: Serial exactly matches the consecutive-operations definition. (PDF pp. 72, 71.)

C - Incorrect: Cascadeless means reading only committed writes. (PDF pp. 72, 71.)

D - Incorrect: Recoverable constrains commit order after a read dependency. (PDF pp. 72, 71.)

E - Incorrect: A listed definition does match, so None is wrong. (PDF pp. 72, 71.)

Primary note proof

Study-note section ADB09

A serial schedule executes each transaction's operations consecutively. A serializable schedule is equivalent to some serial order; it may interleave transactions. Equality of one final value is insufficient evidence because different executions can accidentally coincide. Two operations conflict when they belong to different transactions, access the same item and at least one writes. Read/read pairs do not conflict.

Study-note section ADB10

Recoverable means if Tj reads Ti's value, Tj cannot commit before Ti commits.

Study-note section ADB10

Cascadeless means a transaction reads only values written by committed transactions, avoiding cascading aborts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 72 | lines 6-18

- Serial schedule:
  - A schedule S is serial if, for every transaction Ti participating in the schedule, all the operations of Ti are executed consecutively in the schedule. Otherwise, the schedule is called nonserial schedule.
  - For example, in the banking example suppose there are two transaction where one transaction calculate the interest on the account and another deposit some money into the account. hence the order of execution is important for the final result.
- Serializable schedule:
  - a schedule whose effect on any consistent database instance is identical to that of some complete serial schedule over the set of committed transactions in S.
  - A nonserial schedule S is serializable , is equivalent to say that it is correct to the result of one of the serial schedule .Example,

Fundamental of Database Systems module | PDF page 71 | lines 15-26

- Recoverable schedule:
  - One where no committed transaction needs to be rolled back.
  - A schedule S is recoverable if no transaction Tj in S commits until all transactions Ti that have written an item that Tj reads have committed. Examples,
- Sc: r1(X); w1(X); r2(X); r1(Y);w2(x);c2;a1; not recoverable
- Sd: r1(X); w1(X); r2(X); r1(Y); w2(X);w1(Y); c1; c2;
- Se: r1(X); w1(X); r2(X); r1(Y); w2(x) ; w1(Y); a1; a2;
- Cascadeless schedule:
  - One where every transaction reads only the items that are written by committed transactions. Eg.
- Sf: r1(X); w1(X); r1(Y); c1; r2(X); w2(X);w1(Y); c2;
- Strict Schedules:

Question sources

Addis Ababa University mock exam · Question 18

Question 13 | Q00117

A transaction property state about changes applied to the database by a committed transaction must persist in the database?

A. Isolation

B. Durability or permanency

C. Atomicity

D. Consistency preservation

E. None

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: Durability or permanency

Persisting committed modifications despite subsequent failure is durability or permanency.

Why each choice is right or wrong

A - Incorrect: Isolation regulates visibility between transactions. (PDF pp. 66.)

B - Correct: Durability protects committed changes against loss. (PDF pp. 66.)

C - Incorrect: Atomicity is all-or-nothing execution. (PDF pp. 66.)

D - Incorrect: Consistency preserves valid states rather than persistence. (PDF pp. 66.)

E - Incorrect: Durability is a listed matching property. (PDF pp. 66.)

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 66 | lines 4-13

> • Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
> • Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.
> • Isolation: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary

- Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.

Example

Question sources

Addis Ababa University mock exam · Question 19

Question 14 | Q00118

Which update-timing policy keeps a transaction's uncommitted changes outside the persistent database until commit?

A. Undo

B. In-place modification

C. Immediate-modification

D. Deferred-modification

Hint

Separate update timing from update location and follow the commit boundary.

Answer D: Deferred-modification

Deferred modification keeps uncommitted updates outside the persistent database. If the transaction fails before commit, those database pages do not require undo for its changes; redo can still be needed after commit in a no-force system. Immediate modification permits precommit persistent changes. In-place modification describes overwriting the original location without specifying the commit timing, while undo is an action that reverses applied effects. D therefore names the requested timing policy.

Why each choice is right or wrong

A - Incorrect: Incorrect: undo reverses effects already applied rather than naming a policy postponing them.

B - Incorrect: Incorrect: in-place modification specifies overwriting location, not postponement until commit.

C - Incorrect: Incorrect: immediate modification permits persistent changes before commit.

D - Correct: Correct: deferred modification keeps uncommitted changes outside the persistent database.

Primary note proof

Study-note section ADB29

Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO. Redo-only and NO-UNDO/REDO mean the same thing, so

those cannot be separate unique answers. Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Study-note section ADB29

In-place update overwrites the original location.

Related reading (context, not primary note proof)

Study-note section ADB27

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 107 | lines 16-28

• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:
• A set of transactions records their updates in the log.
• At commit point under WAL scheme these updates are saved on database disk.
• No undo is required because no AFIM is flushed to the disk before a transaction commits.

Fundamental of Database Systems module | PDF page 110 | lines 14-30

◦ Deferred Update:
• All transaction updates are recorded in the local workspace (cache)
• All modified data items in the cache is then written after a transaction ends its execution or after a fixed number of transactions have completed their execution.
• During commit the updates are first recorded on the log and then on the database
• If a transaction fails before reaching its commit point undo is not needed because it didn‘t change the database yet
• If a transaction fails after commit (writing on the log) but before finishing saving to the data base redoing is needed from the log
◦ Immediate Update:
• As soon as a data item is modified in cache, the disk copy is updated.
• These update are first recorded on the log and on the disk by force writing, before the database is updated
• If a transaction fails after recording some change to the database but before reaching its commit point , this will be rolled back

Question sources

Addis Ababa University mock exam · Question 20

Question 15 | Q00119

Which one of the following is false?

A. In object-oriented database, data are stored as collections of rows and tables

B. In relational database, data are perceived by users as tables

C. Object oriented database allows object identification and communication

D. Object oriented database allows reusability of objects

E. None of the above

Hint

Distinguish an object instance from its class, attribute values and persistent identity.

Answer A: The rows-and-tables claim is false for the object-oriented model.

An OODBMS primarily models persistent objects with identity, state and behavior, rather than relational rows and tables. Objects communicate through methods and object references, and inheritance supports reuse. Relational presentation as tables is the contrasting model. Therefore A is the false statement.

Why each choice is right or wrong

A - Correct: This is false: an OODBMS primarily models persistent objects rather than relational rows and tables.

B - Incorrect: Presenting relations as tables describes the relational contrast correctly.

C - Incorrect: Object identity and method-based collaboration are supported object-model concepts.

D - Incorrect: Inheritance and reusable object behavior support reuse, so this is not the false statement.

E - Incorrect: A is false, so none of the above cannot answer this negative question.

Primary note proof

Study-note section ADB01

> An object combines identity, state and behavior. A class specifies the attributes and operations of its instances; an object is one instance. The object identifier (OID) distinguishes objects independently of their attribute values and remains stable over that object's lifetime. Two objects can have equal values and different identities; changing a name does not create a new identity.

Study-note section ADB01

> It does not primarily present objects as rows and tables.

Study-note section ADB01

An object-relational DBMS extends a relational system with richer types; ORM maps programming objects into relational tables.

Related reading (context, not primary note proof)

Study-note section ADB02

Study-note section ADB03

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-14

Basic operations:
Selection (σ) - Selects a subset of rows from relation.
Projection (π) - Deletes unwanted columns from relation.
Cross-product (x) - Allows us to combine two relations.
Set-difference (-) - Tuples in relation 1, but not in relation 2.

Question sources

Addis Ababa University mock exam · Question 21

Question 16 | Q00296

27.----------is a collection of multiple logically related database distributed over a computer network.

A. Data Fragmentation

B. Distributed Database

C. Data Allocation

D. Data Replication

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer B: Distributed Database

A distributed database is a coordinated collection of logically related data across networked sites. Fragmentation divides a relation into row or column subsets; replication creates copies; allocation chooses the sites storing fragments or copies. These are design decisions within a distributed database rather than alternative names for the whole collection. Distribution does not require every fragment to be replicated, and the question's definition identifies the database as a whole.

Why each choice is right or wrong

A - Incorrect: Incorrect: fragmentation splits relations into subsets rather than naming the entire networked collection.

B - Correct: Correct: a distributed database is the coordinated logically related data collection across networked sites.

C - Incorrect: Incorrect: allocation assigns fragments or copies to sites; it is a placement decision.

D - Incorrect: Incorrect: replication creates copies and is a design choice rather than the definition of the collection.

Primary note proof

Study-note section ADB33

A distributed database contains logically related data across networked sites, managed as a coordinated system.

Study-note section ADB36

Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation.

Study-note section ADB37

Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 114 | lines 7-13

• A distributed database (DDB) can be defined as :
- A collection of multiple logically related database distributed over a computer network, and a distributed database management system as a software system that manages a distributed database while making the distribution transparent to the user.
- The physical placement of data (files, relations, etc.) which is not known to the user (distribution transparency).

Fundamental of Database Systems module | PDF page 117 | lines 18-31

I. Data Replication
• The system maintain several identical copies of the relation & store each copy at a different site
• In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead
II. Data Fragmentation
- Split a relation into logically related and correct parts.
- The main reasons for fragmenting a relation are

• Efficiency- data that is not needed by the local applications is not stored
• Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency
- but reconstruction of the whole relation will require accessing data from all sites containing part of the relation

Question sources

Computer Science model exam · Question 27

University of Gondar model exam · Question 27

Question 17 | Q00297

..... Means that a transaction must execute exactly once completely or not at all.

A. Consistency

B. Durability

C. Isolation

D. Atomicity

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer D: Atomicity

Atomicity means a transaction is performed in its entirety or not performed at all. 'Exactly once' is interpreted here as all-or-nothing completion, rather than exactly-once delivery semantics.

Why each choice is right or wrong

A - Incorrect: Consistency concerns valid before/after states. (PDF pp. 66.)

B - Incorrect: Durability concerns survival after commit. (PDF pp. 66.)

C - Incorrect: Isolation concerns interaction with other transactions. (PDF pp. 66.)

D - Correct: Atomicity supplies the all-or-nothing rule. (PDF pp. 66.)

Primary note proof

Study-note section ADB06

A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 66 | lines 4-13

- Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
- Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.
- Isolation: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary
- Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.

Example

Question sources

Computer Science model exam · Question 28

University of Gondar model exam · Question 28

Question 18 | Q00300

Checkpoints are a part of

A. Recovery measures

B. Concurrency measures

C. Authorization measures

D. Security measures

Hint

Identify which history a recovery boundary can safely exclude and whether active work spans it.

Answer A: Recovery measures

A checkpoint synchronizes database and log so recovery need not search the entire history. Its purpose is recovery work reduction.

Why each choice is right or wrong

A - Correct: Checkpointing is described in database recovery. (PDF pp. 105, 107.)

B - Incorrect: Locking and scheduling are concurrency controls; this checkpoint has a recovery role. (PDF pp. 105, 107.)

C - Incorrect: Privileges authorize access rather than synchronize recovery state. (PDF pp. 105, 107.)

D - Incorrect: Checkpointing does not authenticate users or grant privileges. (PDF pp. 105, 107.)

Primary note proof

Study-note section ADB30

A checkpoint records enough durable recovery state to avoid scanning all old history. In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo. Active transactions can span the checkpoint, so their relevant earlier updates can still need undo. Do not assume every transaction before a checkpoint needs no processing.

Study-note section ADB30

Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Study-note section ADB30

Without checkpoints or another valid truncation boundary, long-running systems may need expensive log scans; deferred update does not avoid that problem.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 105 | lines 25-36

Check pointing
• Log file is used to recover failed DB but we may not know how far back in the log to search. Thus
• Checkpoint is a Point of synchronization between database and log file.
• Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.
• When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Fundamental of Database Systems module | PDF page 107 | lines 16-28

• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:
• A set of transactions records their updates in the log.
• At commit point under WAL scheme these updates are saved on database disk.
• No undo is required because no AFIM is flushed to the disk before a transaction commits.

Question sources

Computer Science model exam · Question 31

University of Gondar model exam · Question 31

Question 19 | Q00304

Which of the following is not a recovery technique?

A. ARIES Algorithm

B. Two-phase commit

C. Immediate update

D. Deferred update

Hint

Distinguish coordinated transaction outcome from concurrent access and local storage.

Answer B: Two-phase commit

Deferred update, immediate update, and ARIES are recovery approaches. Two-phase commit coordinates the distributed commit decision; it is the intended exception among the named recovery algorithms, though commit protocols also support distributed recovery.

Why each choice is right or wrong

A - Incorrect: ARIES is explicitly a recovery algorithm. (PDF pp. 107, 109, 123.)

B - Correct: Two-phase commit is a distributed commit approach, not the listed update-recovery algorithm. (PDF pp. 107, 109, 123.)

C - Incorrect: Immediate update appears under recovery techniques. (PDF pp. 107, 109, 123.)

D - Incorrect: Deferred update is a recovery scheme. (PDF pp. 107, 109, 123.)

Primary note proof

Study-note section ADB39

> Two-phase commit (2PC) first asks participants to prepare and durably promise they can commit. If all vote yes, the coordinator records commit durably and tells participants; otherwise it decides abort. A prepared participant must retain enough state to obey the final decision. Coordinator/communication failure can leave it in doubt and block progress until the decision is learned. Timeout after preparation is not permission to invent a commit or abort.

Study-note section ADB39

> 2PC coordinates atomic outcome; locking/other concurrency protocols provide isolation.

Study-note section ADB29

> Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Related reading (context, not primary note proof)

Study-note section ADB31

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 107 | lines 16-28

• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:
• A set of transactions records their updates in the log.
• At commit point under WAL scheme these updates are saved on database disk.
• No undo is required because no AFIM is flushed to the disk before a transaction commits.

Fundamental of Database Systems module | PDF page 109 | lines 13-22

iii. Shadow Paging
• Maintain two page tables during life of a transaction: current page and shadow page table.
• When transaction starts, two pages are the same.
• Shadow page table is never changed thereafter and is used to restore database in event of failure.
• During transaction, current page table records all updates to database.
• When transaction completes, current page table becomes shadow page table.
X and Y: Shadow copies of data items
X' and Y': Current copies of data items

Fundamental of Database Systems module | PDF page 123 | lines 23-29

i. Primary site technique: A single site is designated as a primary site which serves as a coordinator for transaction management.
• Transaction management:
- Concurrency control and commit are managed by this site.
- In two phase locking, this site manages locking and releasing data items. If all transactions follow two-phase policy at all sites, then serializability is guaranteed.
- Advantages:

Question sources

Computer Science model exam · Question 35

University of Gondar model exam · Question 35

Question 20 | Q00305

Which of the following is not true about OODBMS (object-oriented database management system).

A. The key concepts is objects

B. The object is persistent

C. The key concept is relation

D. Object Identity is unchangeable

Hint

Distinguish an object instance from its class, attribute values and persistent identity.

Answer C: Relation is not the defining concept of an OODBMS.

The defining unit of an object database is an object with identity, state and behavior. ADB01 explains both persistence beyond program execution and identifier stability throughout an object lifetime. Relation is instead the characteristic abstraction of the relational model. Thus C is the one statement that fails to describe an OODBMS.

Why each choice is right or wrong

A - Incorrect: Objects are the defining units of the object-oriented database model.

B - Incorrect: An OODBMS provides persistence so objects survive the program execution that created them.

C - Correct: This is false for OODBMS: relation is the central abstraction of the relational model.

D - Incorrect: The object identifier remains stable throughout the lifetime of its object.

Primary note proof

Study-note section ADB01

> An object combines identity, state and behavior. A class specifies the attributes and operations of its instances; an object is one instance. The object identifier (OID) distinguishes objects independently of their attribute values and remains stable over that object's lifetime. Two objects can have equal values and different identities; changing a name does not create a new identity.

Study-note section ADB01

> Persistence lets an object survive beyond the program execution that created it.

Study-note section ADB01

> Complex CAD or multimedia structures can suit an object model, but many-to-many relationships alone do not require abandoning relational design.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 119 | lines 11-19

• Heterogeneous
- At least one of the database must be from different vendor : two variants
- Federated: Each site may run different database system but the data access is managed through a single conceptual schema.
• This implies that the degree of local autonomy is minimum. Each site must adhere to a centralized access policy. There may be a global schema.
- Multidatabase: There is no one conceptual global schema. For data access a schema is constructed dynamically as needed by the application software.
Site 2

Question sources

Computer Science model exam · Question 36

University of Gondar model exam · Question 36

Question 21 | Q00306

Which optimization approach uses algebraic rules, such as pushing suitable selections before joins, to choose an operation order without exhaustively enumerating plans or comparing estimated costs?

A. Semantic query optimization based on trusted constraints

B. Heuristic- based

C. Cost estimation

D. Exhaustive search

Hint

Consider which equivalent transformation changes intermediate work and what assumptions preserve meaning.

Answer B: Heuristic- based

Heuristic optimization orders or rewrites operations using algebraic rules intended to reduce intermediate work. Applying suitable selections early is a typical example. Semantic optimization additionally uses trusted constraints, cost estimation predicts resource usage for candidate strategies, and exhaustive search enumerates alternatives rather than selecting a small set through rules alone. Although practical optimizers can combine these methods, the stated stem asks specifically for rule-based algebraic ordering, making B unique.

Why each choice is right or wrong

A - Incorrect: Incorrect: this alternative bases deductions on trusted constraints rather than the isolated algebraic ordering rules.

B - Correct: Correct: heuristic optimization orders operations through algebraic rules such as early selection.

C - Incorrect: Incorrect: cost estimation predicts candidate-plan resources instead of the specified rule-only choice.

D - Incorrect: Incorrect: exhaustive search enumerates alternatives, which the stem explicitly excludes.

Primary note proof

Study-note section ADB18

> Heuristics reduce intermediate work: push selective row filters toward base tables; replace product followed by a join predicate with a join; project unneeded columns away while retaining join/filter columns; reorder inner joins when equivalence permits; reuse common expressions when worthwhile. Split conjunctive selections when each predicate can be applied earlier to its own input.

Study-note section ADB18

> Semantic optimization additionally uses trusted constraints, such as a key proving a join cannot multiply matching rows. Cost-based optimization estimates candidate plans instead of relying on rules alone.

Related reading (context, not primary note proof)

Study-note section ADB19

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 31 | lines 19-23

> • Heuristics Approach uses the knowledge of the characteristics of the relational algebra operations and the relationship between the operators to optimize the query.
> • Thus the heuristic approach of optimization will make use of:
> ◦ Properties of individual operators
> ◦ Association between operators

Fundamental of Database Systems module | PDF page 34 | lines 5-7

> 2. Apply heuristics rules to optimize the internal representation.
> 3. A query execution plan is generated to execute groups of operations based on the access paths available on the files involved in the query.

Question sources

Computer Science model exam · Question 37

University of Gondar model exam · Question 37

Question 22 | Q00533

A transaction in which either all of the database actions occur or none of them does is called

A. Atomicity

B. Consistency

C. Isolation

D. Durability

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer A: Atomicity

All database actions occurring or none occurring is atomicity, the first ACID property.

Why each choice is right or wrong

A - Correct: Atomicity is the entire-or-none requirement. (PDF pp. 66.)

B - Incorrect: Consistency is preservation of valid states. (PDF pp. 66.)

C - Incorrect: Isolation separates concurrent effects. (PDF pp. 66.)

D - Incorrect: Durability preserves completed commits. (PDF pp. 66.)

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 66 | lines 4-13

> • Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
> • Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.
> • Isolation: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary
> • Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.
> Example

Question sources

Computer Science model exam · 2015 · Question 2

Question 23 | Q00540

A transaction is still active when one of its changes is written into the persistent database. Which update-timing policy permits this?

A. Deferred-Update

B. In-place update

C. Immediate-Update

D. Undo

Hint

Separate update timing from update location and follow the commit boundary.

Answer C: Immediate-Update

Immediate update permits persistent database changes while the transaction is still active, before commit. An abort may therefore require undo. Deferred update keeps uncommitted changes outside the persistent database. In-place update describes overwriting a location rather than the commit-timing policy, and undo is the reversal of applied effects. The question concerns which timing policy permits the observed precommit write, not whether all immediate-update systems must flush every cache change immediately.

Why each choice is right or wrong

A - Incorrect: Incorrect: deferred update excludes uncommitted changes from the persistent database.

B - Incorrect: Incorrect: in-place update describes overwriting the existing location rather than the commit-timing policy.

C - Correct: Correct: immediate update permits persistent changes before commit.

D - Incorrect: Incorrect: undo reverses effects; it is not the update-timing policy.

Primary note proof

Study-note section ADB29

> Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO. Redo-only and NO-UNDO/REDO mean the same thing, so those cannot be separate unique answers. Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Study-note section ADB29

> In-place update overwrites the original location.

Related reading (context, not primary note proof)

Study-note section ADB27

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 110 | lines 14-30

◦ Deferred Update:
• All transaction updates are recorded in the local workspace (cache)
• All modified data items in the cache is then written after a transaction ends its execution or after a fixed number of transactions have completed their execution.
• During commit the updates are first recorded on the log and then on the database
• If a transaction fails before reaching its commit point undo is not needed because it didn‘t change the database yet
• If a transaction fails after commit (writing on the log) but before finishing saving to the data base redoing is needed from the log
◦ Immediate Update:
• As soon as a data item is modified in cache, the disk copy is updated.
• These update are first recorded on the log and on the disk by force writing, before the database is updated
• If a transaction fails after recording some change to the database but before reaching its commit point , this will be rolled back

Fundamental of Database Systems module | PDF page 107 | lines 16-28

• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:
• A set of transactions records their updates in the log.
• At commit point under WAL scheme these updates are saved on database disk.
• No undo is required because no AFIM is flushed to the disk before a transaction commits.

Question sources

Computer Science model exam · 2015 · Question 9

Question 24 | Q00556

Which of the following occurs when a transaction cannot proceed for an indefinite period while other transactions in the system continue normally?

A. Starvation

B. Deadlock

C. checkpoint

D. lock point

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer A: Starvation

Starvation means indefinite postponement of a transaction while other transactions continue. It can result from repeated waiting or repeated victim selection without any waiting cycle. Deadlock instead involves a cycle of mutually blocked transactions. A checkpoint records recovery state, and the lock point in two-phase locking is the instant a transaction acquires its final lock. Neither is the name of indefinite unfair postponement, so A remains the unique answer.

Why each choice is right or wrong

A - Correct: Correct: indefinite postponement while others proceed is starvation.

B - Incorrect: Incorrect: deadlock requires a waiting cycle, which is not required for starvation.

C - Incorrect: Incorrect: a checkpoint records durable recovery state; it is not an indefinite waiting condition.

D - Incorrect: Incorrect: the lock point is the instant the transaction acquires its final lock in 2PL.

Primary note proof

Study-note section ADB13

> Fixed resource ordering and conservative 2PL prevent cycles. A no-wait policy aborts immediately on an incompatible lock request and needs no age timestamp. A timeout aborts long waits but may mistake slow progress for deadlock. Starvation is indefinite postponement while other work proceeds, without a required cycle. Fair queues, sensible victim selection and retaining original transaction timestamps on retry reduce starvation; deadlock freedom alone does not prove fairness.

Study-note section ADB12

> The lock point is the instant the final lock is obtained.

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 90 | lines 12-21

◦ Starvation occurs when a particular transaction consistently waits or restarted and never gets a chance to proceed further while other transaction continue normally
◦ This may occur , if the waiting method for item locking:
• Gave priority for some transaction over others
• Problem in Victim selection algorithm- it is possible that the same transaction may consistently be selected as victim and rolled-back .example In Wound-Wait
◦ Solution
• FIFO
• Allow for transaction that wait for a longer time

Question sources

Computer Science model exam · 2015 · Question 25

Question 25 | Q00565

The atom constructor can represent the following basic atomic values except.

A. Integers

B. Real Numbers

C. Strings

D. Array

Hint

Classify the shape of a value separately from an operation that creates an instance.

Answer D: Array is not an atom constructor.

ADB04 lists integer, real, Boolean and string as examples of basic atomic values. An array instead supplies indexed positions and is a collection constructor, even when a programming variable holds it as one value. Consequently integers, reals and strings fit the atom category while an array is the requested exception.

Why each choice is right or wrong

A - Incorrect: An integer is explicitly listed as a basic atomic value.

B - Incorrect: A real number is explicitly included among atomic values.

C - Incorrect: A string is a basic atomic value under the object-model convention used here.

D - Correct: An array contains indexed positions and is a collection constructor rather than an atomic one.

Primary note proof

Study-note section ADB04

> In a common object model, an object is described by (OID,type constructor,state). An atom is one basic value, such as an integer, real, Boolean or string. A tuple constructor groups named component values into one structured record, such as (Street:'A',City:'B'). It is not a collection of records.

Study-note section ADB04

> An array is not an atomic constructor merely because stored as one programming variable.

Study-note section ADB04

> A set is an unordered collection without duplicates; a bag allows duplicates; a list has order; an array provides indexed positions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 24 | lines 10-14

> Basic operations:
> Selection (σ) - Selects a subset of rows from relation.
> Projection (π) - Deletes unwanted columns from relation.
> Cross-product (x) - Allows us to combine two relations.
> Set-difference (-) - Tuples in relation 1, but not in relation 2.

Question sources

Computer Science model exam · 2015 · Question 34

Question 26 | Q00584

A system crashes again during ARIES recovery. Durable WAL, pageLSNs and compensation log records survive; storage is intact. No checkpoints were taken. Which statement correctly describes the second restart?

A. The same undo and redo list will be used while recovering again

B. The database will become inconsistent

C. Durable compensation records and pageLSNs preserve recovery progress, so already-installed actions can be skipped or safely redone as required.

D. The system cannot any further

Hint

Trace durable recovery metadata and what progress survives a second crash.

Answer C: C. Durable compensation records and pageLSNs preserve recovery progress, so already-installed actions can be skipped or safely redone as required.

ARIES records recovery progress in durable metadata. PageLSNs help determine whether a redo action is already installed, and compensation records preserve undo progress and its continuation. A second restart can therefore recover safely without assuming that the same undo and redo work lists recur. Lack of checkpoints may increase scanning work, but it does not erase this progress or make further recovery impossible.

Why each choice is right or wrong

A - Incorrect: Durable progress can change the required work, so identical lists are not guaranteed.

B - Incorrect: Restartable recovery does not make inconsistency inevitable.

C - Correct: PageLSNs and durable CLRs support safe continuation after repeated crashes.

D - Incorrect: ARIES can restart using surviving durable recovery metadata.

Primary note proof

Study-note section ADB31

> ARIES uses WAL, page log sequence numbers (pageLSN), a transaction table and a dirty-page table. Analysis reconstructs active/loser transactions and dirty-page information. Redo repeats history, starting from the relevant earliest dirty-page update; pageLSNs help skip already-installed actions. Undo reverses loser transactions.

Study-note section ADB31

> This makes recovery restartable: after a second crash, durable undo progress need not be undone again as if no progress occurred.

Study-note section ADB31

> Compensation log records (CLRs) describe undo actions and their continuation information.

Related reading (context, not primary note proof)

Study-note section ADB27

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 108 | lines 24-29

> • During recovery, all transactions of the commit table are redone and all transactions of active tables are ignored since none of their AFIMs reached the database.
> • It is possible that a commit table transaction may be redone twice but this does not create any inconsistency because of a redone is —idempotent‖,
> • that is, one redone for an AFIM is equivalent to multiple redone for the same AFIM.

Fundamental of Database Systems module | PDF page 110 | lines 14-30

◦ Deferred Update:
• All transaction updates are recorded in the local workspace (cache)
• All modified data items in the cache is then written after a transaction ends its execution or after a fixed number of transactions have completed their execution.
• During commit the updates are first recorded on the log and then on the database
• If a transaction fails before reaching its commit point undo is not needed because it didn‘t change the database yet
• If a transaction fails after commit (writing on the log) but before finishing saving to the data base redoing is needed from the log
◦ Immediate Update:
• As soon as a data item is modified in cache, the disk copy is updated.
• These update are first recorded on the log and on the disk by force writing, before the database is updated
• If a transaction fails after recording some change to the database but before reaching its commit point , this will be rolled back

Question sources

Computer Science model exam · 2015 · Question 53

Question 27 | Q00590

The rigorous two-phase locking protocol permits releasing all locks at the_____

A. Beginning of the transaction

B. During the execution of transaction

C. End of transaction

D. Never in the life-time of transaction

Hint

Track the first lock release and the transaction-completion boundary.

Answer C: At transaction end, on commit or abort.

Rigorous 2PL retains both shared and exclusive locks until commit or abort, unlike basic 2PL, which permits release in its shrinking phase. FDB-style early release would allow interference before completion and would not meet rigorous locking. The locks are eventually released at the transaction boundary, so never is also false.

Why each choice is right or wrong

A - Incorrect: Releasing locks at the beginning would not protect the transaction's later accesses.

B - Incorrect: Rigorous 2PL does not permit early release during ongoing transaction execution.

C - Correct: Both shared and exclusive locks remain held until commit or abort ends the transaction.

D - Incorrect: Locks are released at completion; the protocol does not hold them forever.

Primary note proof

Study-note section ADB12

Strict 2PL retains exclusive locks until commit/abort. Rigorous 2PL retains both shared and exclusive locks until that point. Conservative/static 2PL obtains all needed locks before execution and does not begin while only a subset is available, preventing hold-and-wait deadlock. It requires knowing the lock set and can reduce concurrency. Two-phase locking orders access to data; two-phase commit coordinates a distributed commit decision.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 86 | lines 16-24

◦ Locking (Growing) Phase:
• A transaction applies locks (read or write) on desired data items one at a time.
• acquires all locks but cannot release any locks.
◦ Unlocking (Shrinking) Phase:
• A transaction unlocks its locked data items one at a time.
• Releases locks but cannot acquire any new locks.
● Requirement:
◦ For a transaction these two phases must be mutually exclusively, that is, during

Fundamental of Database Systems module | PDF page 94 | lines 13-20

4.2.5 Validation (Optimistic) Concurrency Control Schemes
◦ This technique allow transaction to proceed asynchronously and only at the time of commit, serializability is checked &
◦ transactions are aborted in case of non-serializable schedules.
◦ Good if there is little interference among transaction
◦ It has three phases: Read, Validation , and Write phase
i.. Read phase:
◦ A transaction can read values of committed data items. However, updates are

Question sources

Computer Science model exam · 2015 · Question 59

Question 28 | Q00594

In which of the following state of the transaction executes all its operations successfully and all the effects are now permanently saved on the database system?

A. Active

B. Partial committed

C. Committed

D. Failed

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer C: Committed

The committed state guarantees persistent changes. Partially committed only signals the end of read/write operations and does not yet ensure permanence.

Why each choice is right or wrong

A - Incorrect: Active means execution is in progress. (PDF pp. 65.)

B - Incorrect: Partially committed explicitly does not guarantee permanent modification. (PDF pp. 65.)

C - Correct: Committed ensures persistent changes. (PDF pp. 65.)

D - Incorrect: Failed is an aborted or unsuccessful transaction. (PDF pp. 65.)

Primary note proof

Study-note section ADB06

> Active means operations are executing. Partially committed means the last operation finished but durable commitment is not yet assured. Committed means commitment succeeded. A failure leads to Failed, then Aborted after rollback; a transaction may terminate or be restarted as a new attempt. Concurrency control primarily enforces isolation; recovery supports atomicity and durability. Transaction management is the broader subsystem, so both it and recovery management can be legitimate answers to broad responsibility questions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 65 | lines 4-7

> ◦ Committed state -ensures that all the changes done on a record by a transition were done persistently
> ◦ Failed state happens when a transaction is aborted during its active state or if one of the rechecking is fails

Question sources

Computer Science model exam · 2015 · Question 63

Question 29 | Q00598

Let transaction T1 has obtained a shared mode lock S on data item Y and transaction T2 has obtained an exclusive lock X on data item Z , according to the above statement which one is correct?

A. T1 can read Y but cannot write Y

B. T2 can read Z but cannot write Z

C. T1 can read Z but cannot write Z

D. T2 can read and write item Y

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer A: T1 can read Y but cannot write Y

Shared locks permit reads but not updates. Exclusive locks permit both reading and updating while excluding other transactions' locks on that item.

Why each choice is right or wrong

A - Correct: T1's shared lock authorizes reading Y, not writing it. (PDF pp. 82, 83.)

B - Incorrect: T2's exclusive lock permits both read and update of Z. (PDF pp. 82, 83.)

C - Incorrect: T1 cannot read Z while another transaction holds its exclusive lock. (PDF pp. 82, 83.)

D - Incorrect: T2 cannot obtain an exclusive write lock on Y while T1 holds its shared lock. (PDF pp. 82, 83.)

Primary note proof

Study-note section ADB11

> A shared (S) lock permits reading; multiple S locks on the same item can coexist. An exclusive (X) lock permits its holder to read and write and conflicts with another S or X on that item. Locks on different items do not conflict merely because one is exclusive. These are the basic lock-based rules; multiversion systems can permit additional snapshot-reader behavior.

Study-note section ADB11

> A transaction must own the appropriate lock before access and can release only its own locks.

Study-note section ADB11

> One transaction may request a lock on an already locked item: it is granted if compatible, otherwise waits or aborts under the policy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 82 | lines 20-26

• Shared mode: shared lock (X)
• More than one transaction can apply share lock on X for reading its value but no write lock can be applied on X by any other transaction.
• Exclusive mode: Write lock (X)
• Only one write lock on X can exist at any time and no shared lock can be applied by any other transaction on X.
• Conflict matrix

Fundamental of Database Systems module | PDF page 83 | lines 2-8

• Lock Manager:
• Managing locks on data items.
• Lock table:
• Lock manager uses it to store the identify of transaction locking a data item, the data item, lock mode and pointer to the next data item locked. One simple way to implement a lock table is through linked list.

Question sources

Computer Science model exam · 2015 · Question 67

Question 30 | Q00615

Which claim about deadlock prevention and starvation is false?

A. In conservative 2PL, starvation may occur

B. Wound-wait guarantees starvation freedom regardless of restart priorities or scheduling fairness

C. Deadlock can be prevented by wait-die

D. Some deadlock-prevention methods may cause starvation

Hint

Identify requester and holder ages before applying the protocol's action.

Answer B: Wound-wait guarantees starvation freedom regardless of restart priorities or scheduling fairness

Deadlock freedom does not itself guarantee that each transaction eventually completes. Wound-wait uses transaction age to decide whether a requester waits or aborts a younger holder, but a starvation-freedom argument also needs appropriate restart priorities, fair scheduling and holders that eventually finish. Its unconditional guarantee in B is therefore false. Conservative 2PL can postpone a transaction indefinitely under unfair admission, and wait-die prevents waiting cycles. Some prevention policies can cause repeated waiting or victimization.

Why each choice is right or wrong

A - Incorrect: True: conservative 2PL prevents cycles but does not by itself guarantee fair admission.

B - Correct: Correct selection: the unconditional starvation-freedom claim omits necessary progress and priority assumptions.

C - Incorrect: True: wait-die is a timestamp-based deadlock-prevention policy.

D - Incorrect: True: preventing deadlock can still leave a transaction repeatedly postponed or aborted.

Primary note proof

Study-note section ADB14

> Retain the original timestamp when restarting an aborted transaction so it gains relative priority as new transactions arrive. Starvation avoidance additionally assumes holders eventually finish and scheduling is fair.

Study-note section ADB13

> Fixed resource ordering and conservative 2PL prevent cycles. A no-wait policy aborts immediately on an incompatible lock request and needs no age timestamp. A timeout aborts long waits but may mistake slow progress for deadlock. Starvation is indefinite postponement while other work proceeds, without a required cycle. Fair queues, sensible victim selection and retaining original transaction timestamps on retry reduce starvation; deadlock freedom alone does not prove fairness.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 88 | lines 19-27

> ● Wait-die
> ◦ If Ti has higher priority, it is allowed to wait; otherwise it is aborted.
> ◦ An older transaction is allowed to wait on a younger transaction.
> ◦ A younger transaction requesting an item held by an older transaction is aborted
> ◦ If TS(Ti) < TS(Tj), then (Ti older than Tj)Ti is allowed to wait.
> ◦ Otherwise (Ti younger than Tj)Abort Ti (Ti dies) and restart it later with the same timestamp
> T1(ts =10): M is locked , K is requested
> T2 (ts =20): K is locked, Z is requested

Fundamental of Database Systems module | PDF page 89 | lines 4-13

> • Wound-wait
> - The opposite of wait-die
> - If Ti has higher priority, abort Tj; otherwise Ti waits.
> - A younger transaction is allowed to wait on an older one
> - An older transaction requesting an item held by a younger transaction preempts the younger transaction by aborting it.
> - If TS(Ti) < TS(Tj), then (Ti older than Tj),Abort Tj (Ti wounds Tj) and restart Tj later with the same timestamp
> - Otherwise (Ti younger than Tj)Ti is allowed to wait
> T2 (ts =20)

Fundamental of Database Systems module | PDF page 90 | lines 12-21

◦ Starvation occurs when a particular transaction consistently waits or restarted and never gets a chance to proceed further while other transaction continue normally
◦ This may occur , if the waiting method for item locking:
• Gave priority for some transaction over others
• Problem in Victim selection algorithm- it is possible that the same transaction may consistently be selected as victim and rolled-back
.example In Wound-Wait
◦ Solution
• FIFO
• Allow for transaction that wait for a longer time

Question sources

Computer Science model exam · 2015 · Question 84

Question 31 | Q00616

Concurrency control in DBMS is important for which of the following reasons?

A. To ensure data integrity when updates occur to the database in a single user environment

B. To ensure data integrity when reads occur to the database in a single user environment

C. To ensure data integrity when updates occur to the database in a multi user environment

D. To ensure data integrity when reads occur to the database in a multi user environment

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer C: To ensure data integrity when updates occur to the database in a multi user environment

Concurrency control preserves consistency when multiple users' updates interleave and conflict. Read-only sharing alone does not cause lost updates; the intended scenario is a multiuser system with updates.

Why each choice is right or wrong

A - Incorrect: Single-user updates lack concurrent intertransaction conflicts. (PDF pp. 68, 70, 79.)

B - Incorrect: Single-user reads do not create the cited conflicts. (PDF pp. 68, 70, 79.)

C - Correct: Multiuser updates can create read/write and write/write conflicts requiring scheduling. (PDF pp. 68, 70, 79.)

D - Incorrect: Reads can be involved when others update, but reads alone do not capture the source's motivating condition. (PDF pp. 68, 70, 79.)

Primary note proof

Study-note section ADB08

Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Study-note section ADB06

Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure.

Study-note section ADB09

Read/read pairs do not conflict.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 68 | lines 4-7

i. The Lost Update Problem
◦ This occurs when two transactions that access the same database items have their operations interleaved in a way that makes the value of some database item incorrect.

Fundamental of Database Systems module | PDF page 70 | lines 2-12

ii. The Temporary Update (or Dirty Read) Problem
◦ This occurs when one transaction updates a database item and then the transaction fails for some reason .
◦ The updated item is accessed by another transaction before it is changed back to its original value. Based on the above scenario:
iii. The Incorrect Summary Problem
◦ If one transaction is calculating an aggregate summary function on a number of records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.
Example: T1 would like to add the values of A=10, B=20 and C=30. after the values are read by

Fundamental of Database Systems module | PDF page 79 | lines 4-15

- Transaction Processor is divided into:
◦ A concurrency-control manager, or scheduler, responsible for assuring isolation of transactions
◦ A logging and recovery manager, responsible for the durability of transactions.
- The scheduler (concurrency-control manager) must assure that the individual actions of multiple transactions are executed in such an order that the net effect is the same as if the transactions had in fact executed one-at-a-time.

1. Purpose of Concurrency Control

◦ To enforce Isolation (through mutual exclusion) among conflicting transactions.
◦ To preserve database consistency through consistency preserving execution of transactions.
◦ To resolve read-write and write-write conflicts.

Question sources

Computer Science model exam · 2015 · Question 85

Question 32 | Q00624

A lock wait-for graph may contain several independent deadlock cycles. Which termination methods guarantee eliminating all deadlock cycles? Assume each aborted transaction is undone and releases its locks. Select all that apply. 'Half' and 'single' mean arbitrary selections without a repeated cycle check.

A. Abort all deadlocked transaction

B. Abort half of the deadlocked transaction

C. Aborting single deadlocked transactio

D. Aborting one transaction at a time until the deadlock cycle is eliminated

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer A, D: A. Abort all deadlocked transaction; D. Aborting one transaction at a time until the deadlock cycle is eliminated

Aborting all deadlocked transactions removes every participating wait, although it wastes work. A more selective complete method aborts a suitable victim, releases its locks and repeats until no cycles remain. An arbitrary half of the participants can leave a cycle intact, and one arbitrary victim may resolve only one of several independent cycles. The guarantee therefore applies to A and D.

Why each choice is right or wrong

A - Correct: Removing all deadlocked participants eliminates all their cycles.

B - Incorrect: An arbitrary half may miss an entire remaining cycle.

C - Incorrect: One victim need not break other independent cycles.

D - Correct: Repeated victim selection and graph checking continues until every cycle is gone.

Primary note proof

Study-note section ADB13

Deadlock requires a waiting cycle: T1 holds X and wants Y; T2 holds Y and wants X. In a wait-for graph, an edge Ti→Tj means Ti waits for a lock held by Tj. A cycle indicates deadlock under the ordinary lock-wait model. Abort a suitable victim, undo its effects and release its locks; repeat if

other cycles remain. Aborting every deadlocked transaction also breaks cycles but wastes more work.

Study-note section ADB13

A no-wait policy aborts immediately on an incompatible lock request and needs no age timestamp.

Study-note section ADB13

Starvation is indefinite postponement while other work proceeds, without a required cycle.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 89 | lines 4-13

• Wound-wait
- The opposite of wait-die
- If Ti has higher priority, abort Tj; otherwise Ti waits.
- A younger transaction is allowed to wait on an older one
- An older transaction requesting an item held by a younger transaction preempts the younger transaction by aborting it.
- If TS(Ti) < TS(Tj), then (Ti older than Tj),Abort Tj (Ti wounds Tj) and restart Tj later with the same timestamp
- Otherwise (Ti younger than Tj)Ti is allowed to wait
T2 (ts =20)

Fundamental of Database Systems module | PDF page 90 | lines 12-21

◦ Starvation occurs when a particular transaction consistently waits or restarted and never gets a chance to proceed further while other transaction continue normally
◦ This may occur , if the waiting method for item locking:
• Gave priority for some transaction over others
• Problem in Victim selection algorithm- it is possible that the same transaction may consistently be selected as victim and rolled-back
.example In Wound-Wait
◦ Solution
• FIFO
• Allow for transaction that wait for a longer time

Question sources

Computer Science model exam · 2015 · Question 93

Question 33 | Q00643

Among the components of the DBMS, which one is responsible for keeping the isolation of transactions?

A. Recovery protocol

B. Application programmer

C. Concurrency -control protocol

D. Database administrator

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer C: Concurrency -control protocol

Concurrency control regulates interactions between simultaneous transactions and enforces isolation. Recovery supports atomicity and durability by undoing incomplete effects and preserving committed work. Application programmers must supply correct transaction logic, and the DBA administers the database rather than serving as the executing concurrency-control mechanism. The question asks for that mechanism's primary responsibility, so C is the precise answer even though administration, applications and recovery all contribute to a working database.

Why each choice is right or wrong

A - Incorrect: Incorrect: recovery primarily supports atomicity and durability, rather than scheduling concurrent interactions for isolation.

B - Incorrect: Incorrect: an application programmer supplies transaction logic but is not the executing isolation-control mechanism.

C - Correct: Correct: concurrency control regulates conflicting concurrent operations to enforce isolation.

D - Incorrect: Incorrect: the DBA is an administrative human role, not the DBMS mechanism scheduling transaction interactions.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Study-note section ADB06

> Active means operations are executing. Partially committed means the last operation finished but durable commitment is not yet assured. Committed means commitment succeeded. A failure leads to Failed, then Aborted after rollback; a transaction may terminate or be restarted as a new attempt. Concurrency control primarily enforces isolation; recovery supports atomicity and

durability. Transaction management is the broader subsystem, so both it and recovery management can be legitimate answers to broad responsibility questions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 66 | lines 4-13

- Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
- Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.
- Isolation: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary
- Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.

Example

Fundamental of Database Systems module | PDF page 79 | lines 4-15

- Transaction Processor is divided into:
  - A concurrency-control manager, or scheduler, responsible for assuring isolation of transactions
  - A logging and recovery manager, responsible for the durability of transactions.
- The scheduler (concurrency-control manager) must assure that the individual actions of multiple transactions are executed in such an order that the net effect is the same as if the transactions had in fact executed one-at-a-time.

1. Purpose of Concurrency Control
   - To enforce Isolation (through mutual exclusion) among conflicting transactions.
   - To preserve database consistency through consistency preserving execution of transactions.
   - To resolve read-write and write-write conflicts.

Question sources

Computer Science model exam · 2015 · Question 112

Question 34 | Q00645

_____Occurs when two transactions update the same data item, but both read the same original value before update

A. Dirty read

B. Lost update

C. Incorrect summary

D. Unrepeatable read

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer B: Lost update

Both transactions reading the original value lets a later write overwrite the earlier transaction's update; that update is lost.

Why each choice is right or wrong

A - Incorrect: Dirty read concerns uncommitted values from another transaction. (PDF pp. 68, 70.)

B - Correct: Lost update matches the stale-original-value overwrite. (PDF pp. 68, 70.)

C - Incorrect: Incorrect summary combines differently timed values in an aggregate. (PDF pp. 68, 70.)

D - Incorrect: Unrepeatable read concerns a later reread seeing a changed value. (PDF pp. 68, 70.)

Primary note proof

Study-note section ADB08

> Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 68 | lines 4-7

> i. The Lost Update Problem
> ◦ This occurs when two transactions that access the same database items have their operations interleaved in a way that makes the value of some database item incorrect.

Fundamental of Database Systems module | PDF page 70 | lines 2-12

> ii. The Temporary Update (or Dirty Read) Problem
> ◦ This occurs when one transaction updates a database item and then the transaction fails for some reason .
> ◦ The updated item is accessed by another transaction before it is changed back to its original value. Based on the above scenario:
> iii. The Incorrect Summary Problem
> ◦ If one transaction is calculating an aggregate summary function on a number of records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.
> Example: T1 would like to add the values of A=10, B=20 and C=30. after the values are read by

Question sources

Computer Science model exam · 2015 · Question 114

Question 35 | Q00659

Which of the DBMS component ensures that the concurrent execution of multiple operations on the database results into consistent database state?

A. Log

B. Buffer manager

C. File manager

D. Transaction processing system

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer D: Transaction processing system

The transaction processing system contains the scheduler and recovery managers. Its scheduling ensures concurrent individual actions have a net effect equivalent to one-at-a-time execution.

Why each choice is right or wrong

A - Incorrect: The log records changes but does not itself schedule consistent execution. (PDF pp. 79, 99.)

B - Incorrect: Buffers hold blocks and arrange writes, not the entire consistency protocol. (PDF pp. 79, 99.)

C - Incorrect: File management is not the scheduler responsible for concurrent effects. (PDF pp. 79, 99.)

D - Correct: Transaction processing contains the concurrency-control component implementing this guarantee. (PDF pp. 79, 99.)

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Study-note section ADB09

> A serial schedule executes each transaction's operations consecutively.

Study-note section ADB09

> A serializable schedule is equivalent to some serial order; it may interleave transactions.

Related reading (context, not primary note proof)

Study-note section ADB07

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 79 | lines 4-15

- Transaction Processor is divided into:
  - A concurrency-control manager, or scheduler, responsible for assuring isolation of transactions
  - A logging and recovery manager, responsible for the durability of transactions.
- The scheduler (concurrency-control manager) must assure that the individual actions of multiple transactions are executed in such an order that the net effect is the same as if the transactions had in fact executed one-at-a-time.

1. Purpose of Concurrency Control
   - To enforce Isolation (through mutual exclusion) among conflicting transactions.
   - To preserve database consistency through consistency preserving execution of transactions.
   - To resolve read-write and write-write conflicts.

Fundamental of Database Systems module | PDF page 99 | lines 2-8

✓ Logging: In order to assure durability, every change in the database is logged separately on disk.
✓ Log manager initially writes the log in buffers and negotiates with the buffer manager to make sure that buffers are written to disk at appropriate times.
✓ Recovery Manager: will be able to examine the log of changes and restore the database to some consistent state.
Recovery techniques and facilities

Question sources

Computer Science model exam · 2015 · Question 128

Question 36 | Q00687

How do you prevent system privileges deletion and updating authority on PRICE and YEAR column of the CATALOGUE table from user Mohammed

A. GRANT DELETE, UPDATE (price,year) ON catalogue From Mohammed

B. ALLOW DELETE, UPDATE (price,year) ON catalogue From Mohammed

C. REVOKE DELETE, UPDATE (price,year) ON catalogue From Mohammed

D. DENY DELETE, UPDATE (price,year) ON catalogue From Mohammed

Hint

Track whether a privilege is being added or withdrawn and whether it applies to rows or columns.

Answer C: REVOKE DELETE, UPDATE (price,year) ON catalogue From Mohammed

To withdraw Mohammed's DELETE and column-specific UPDATE privileges, use REVOKE DELETE, UPDATE(price,year) ON catalogue FROM Mohammed. This concerns object privileges, despite the stem's 'system privileges' wording. Other independent grants may remain.

Why each choice is right or wrong

A - Incorrect: GRANT adds privileges and also uses TO, not FROM. (PDF pp. 54, 55.)

B - Incorrect: ALLOW is not the module's SQL privilege command. (PDF pp. 54, 55.)

C - Correct: REVOKE with ON and FROM matches the module syntax. (PDF pp. 54, 55.)

D - Incorrect: DENY is not the revocation command presented in this module. (PDF pp. 54, 55.)

Primary note proof

Study-note section ADB25

> GRANT SELECT ON Catalogue TO Mohammed; permits reads. REVOKE DELETE, UPDATE (Price,Year) ON Catalogue FROM Mohammed; removes the specified table deletion and column update grants in SQL dialects supporting that form. DELETE applies to rows/table access rather than particular columns; UPDATE can be column-scoped. REVOKE removes permission, whereas GRANT gives it.

Study-note section ADB25

> WITH GRANT OPTION permits a recipient to grant a privilege onward; removing an upstream grant can cascade or be restricted according to the chosen command and DBMS.

Study-note section ADB25

> Removing one direct grant may leave effective access through another role or grant path.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 54 | lines 12-20

> GRANT <privileges>
> ON <object>
> TO <users>
> [WITH GRANT OPTION]
> • <privileges> is a list of
> SELECT <columns>, INSERT <columns>, DELETE, and UPDATE <columns>, or
> simply ALL
> • <users> is a list of user names or PUBLIC
> • <object> is the name of a table or view

Fundamental of Database Systems module | PDF page 55 | lines 8-12

> REVOKE <privileges>
> ON <object>
> FROM <users>

- If a user has the same privilege from other users then they keep it
- All privileges dependent on the revoked one are also revoked

Question sources

Computer Science model exam 2 · Question 28

Question 37 | Q00890

Inclient-server database architecture, what is the primary role of the server?

A. Provide the GUI for end users

B. Initiate all network connections and poll clients for updates

C. Store and manage shared data, enforce ACID properties, and process client requests

D. Cache all user sessions locally for offline access

Hint

Identify which tier submits, mediates or executes a request.

Answer C: Store and manage shared data, enforce ACID properties, and process client requests

The database server provides database functionality and manages local data, while clients submit requests and combine results. DBMS components enforce transaction properties.

Why each choice is right or wrong

A - Incorrect: The client-facing GUI is not the server's primary data-management role. (PDF pp. 125, 126, 66.)

B - Incorrect: Clients initiate requests; polling clients is not the described workflow. (PDF pp. 125, 126, 66.)

C - Correct: Managing data and processing client requests matches server responsibilities. (PDF pp. 125, 126, 66.)

D - Incorrect: Offline local session caching is not the described database server function. (PDF pp. 125, 126, 66.)

Primary note proof

Study-note section ADB34

In client/server architecture, clients submit requests while servers manage shared data, execute operations and enforce supported integrity/transaction rules. In collaborating-server designs, servers cooperate on subqueries and transactions. Middleware can translate requests, locate data, coordinate operations and combine results across independent servers. These describe communication responsibilities, not fragmentation types.

Study-note section ADB34

State how much schema, transaction and administrative autonomy local sites retain.

Study-note section ADB34

A tightly integrated distributed system can present one global schema and shared control.

Related reading (context, not primary note proof)

Study-note section ADB06

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 125 | lines 11-16

6.6 Client-Server Database Architecture
• It consists of clients running client software, a set of servers which provide all database functionalities and a reliable communication infrastructure.
three-tier client/server architecture.
• Many Web applications use an architecture called the three-tier architecture, which adds an intermediate layer between the client and the database server. This intermediate layer

Fundamental of Database Systems module | PDF page 126 | lines 8-16

• The server software is responsible for local data management at a site, much like centralized DBMS software.
• The client software is responsible for most of the distribution function.
• The communication software manages communication among clients and servers.
• The processing of a SQL queries goes as follows:
• Client parses a user query and decomposes it into a number of independent sub-queries. Each subquery is sent to appropriate site for execution.
• Each server processes its query and sends the result to the client.
• The client combines the results of subqueries and produces the final result.

Fundamental of Database Systems module | PDF page 66 | lines 4-13

- Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
- Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.
- Isolation: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary
- Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.

Example

Question sources

Computer Science Exit Exam · 2018 · Question 3

Question 38 | Q00892

In two-phase locking (2PL), during which phase may a transaction acquire locks but not release any?

A. Growing phase

B. Commit phase

C. .

Validation phase

D. Shrinking phase

Hint

Track the first lock release and the transaction-completion boundary.

Answer A: Growing phase

The growing phase acquires locks without releasing any. Shrinking starts with release and prohibits new acquisitions.

Why each choice is right or wrong

A - Correct: Growing has the acquisition-without-release rule. (PDF pp. 86.)

B - Incorrect: Commit is not the phase name in basic 2PL. (PDF pp. 86.)

C - Incorrect: Validation belongs to optimistic concurrency control. (PDF pp. 86.)

D - Incorrect: Shrinking permits release, not acquisition. (PDF pp. 86.)

Primary note proof

Study-note section ADB12

> Basic 2PL has a growing phase that acquires/upgrades locks and a shrinking phase that releases/downgrades them. Once any lock is released, no new lock may be acquired. The lock point is the instant the final lock is obtained. 2PL guarantees conflict serializability for well-formed transactions, but can deadlock and basic 2PL alone need not prevent cascading aborts.

Study-note section ADB12

> Two-phase locking orders access to data; two-phase commit coordinates a distributed commit decision.

Study-note section ADB16

> Optimistic/validation control reads and computes privately, validates its read/write sets against overlapping transactions, then publishes if validation succeeds; otherwise it retries.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 86 | lines 16-24

◦ Locking (Growing) Phase:
• A transaction applies locks (read or write) on desired data items one at a time.
• acquires all locks but cannot release any locks.
◦ Unlocking (Shrinking) Phase:
• A transaction unlocks its locked data items one at a time.
• Releases locks but cannot acquire any new locks.
● Requirement:
◦ For a transaction these two phases must be mutually exclusively, that is, during

Question sources

Computer Science Exit Exam · 2018 · Question 5

Question 39 | Q00915

Assume durable logs and serializable concurrency control, but no checkpoints or other valid log-truncation boundary. In a high-transaction system using deferred update, which recovery cost can grow after long uptime?

A. Recovery may require scanning a very long accumulated log

B. Increased disk I/O due to frequent page copying

C. Shared locks cannot be used

D. View serializability cannot be ensured

Hint

Identify which history a recovery boundary can safely exclude and whether active work spans it.

Answer A: Recovery may require scanning a very long accumulated log

Checkpoints record recovery state so recovery need not inspect the entire accumulated history. Without a checkpoint or another valid truncation boundary, the log that must be searched can grow throughout a long-running workload. Deferred update does not remove that problem. It specifies when persistent changes occur, not a requirement to copy shadow pages or a prohibition on shared locks. Serializability is enforced by the assumed concurrency-control mechanism, rather than by the existence of checkpoints.

Why each choice is right or wrong

A - Correct: Correct: without a recovery boundary, a long-running system may need to inspect a very large accumulated log.

B - Incorrect: Incorrect: deferred update describes timing and does not require frequent shadow-page copying.

C - Incorrect: Incorrect: deferring persistent writes does not prohibit shared-lock concurrency control.

D - Incorrect: Incorrect: serializability is enforced by the assumed concurrency control; a checkpoint limits recovery work rather than determining schedule equivalence.

Primary note proof

Study-note section ADB30

A fuzzy checkpoint permits ongoing transactions and records active-transaction and dirty-page information without forcing all pages simultaneously. Recovery may start analysis at the checkpoint but redo from an earlier required log position. A checkpoint is neither a full backup nor an automatic commit of active transactions. Without checkpoints or another valid truncation boundary, long-running systems may need expensive log scans; deferred update does not avoid that problem.

Study-note section ADB29

Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO.

Study-note section ADB12

Basic 2PL has a growing phase that acquires/upgrades locks and a shrinking phase that releases/downgrades them. Once any lock is released, no new lock may be acquired. The lock point is the instant the final lock is obtained. 2PL guarantees conflict serializability for well-formed transactions, but can deadlock and basic 2PL alone need not prevent cascading aborts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 105 | lines 25-36

Check pointing
• Log file is used to recover failed DB but we may not know how far back in the log to search. Thus
• Checkpoint is a Point of synchronization between database and log file.
• Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.
• When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Fundamental of Database Systems module | PDF page 108 | lines 24-29

• During recovery, all transactions of the commit table are redone and all transactions of active tables are ignored since none of their AFIMs reached the database.
• It is possible that a commit table transaction may be redone twice but this does not create any inconsistency because of a redone is —idempotent‖,
• that is, one redone for an AFIM is equivalent to multiple redone for the same AFIM.

Question sources

Computer Science Exit Exam · 2018 · Question 28

Question 40 | Q00933

Which listed techniques use transaction logs for recovery in the course model? Interpret deferred modification as the conventional logged, no-force deferred-update method. Select all that apply.

A. Recovery in Multi-database Systems

B. ARIES Recovery Algorithm

C. Shadow Paging Technique

D. Deferred Database Modification

Hint

Separate update timing from update location and follow the commit boundary.

Answer B, D: B. ARIES Recovery Algorithm; D. Deferred Database Modification

ARIES uses write-ahead logging and log-driven recovery metadata. Conventional logged deferred modification also uses recorded updates to redo committed effects that have not reached persistent pages. Both are log-based techniques under the stated model. Shadow paging instead preserves committed page mappings and installs a new map at commit. Recovery in multidatabase systems names an environment rather than one particular algorithm.

Why each choice is right or wrong

A - Incorrect: Multidatabase recovery names an environment rather than identifying a particular log-based technique.

B - Correct: ARIES explicitly relies on WAL and logged recovery metadata.

C - Incorrect: Shadow paging preserves page maps as an alternative to conventional log-based recovery.

D - Correct: Deferred modification can use a log to redo committed changes without undo.

Primary note proof

Study-note section ADB29

> Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO. Redo-only and NO-UNDO/REDO mean the same thing, so those cannot be separate unique answers. Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Study-note section ADB31

> ARIES uses WAL, page log sequence numbers (pageLSN), a transaction table and a dirty-page table.

Study-note section ADB29

Shadow paging keeps an unchanged committed page map and a modified map; commit durably installs a new root/map pointer atomically after required pages are durable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 104 | lines 19-25

• For Undo: Before a data item‘s AFIM is flushed to the database disk (overwriting the BFIM) its BFIM must be written to the log and the log must be saved on a stable store (log disk).
• For Redo: Before a transaction executes its commit operation, all its AFIMs must be written to the log and the log must be saved on a stable store.
• Standard Recovery Terminology

Fundamental of Database Systems module | PDF page 107 | lines 16-28

• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:
• A set of transactions records their updates in the log.
• At commit point under WAL scheme these updates are saved on database disk.
• No undo is required because no AFIM is flushed to the disk before a transaction commits.

Fundamental of Database Systems module | PDF page 109 | lines 13-22

iii. Shadow Paging
• Maintain two page tables during life of a transaction: current page and shadow page table.
• When transaction starts, two pages are the same.
• Shadow page table is never changed thereafter and is used to restore database in event of failure.
• During transaction, current page table records all updates to database.
• When transaction completes, current page table becomes shadow page table.
X and Y: Shadow copies of data items
X' and Y': Current copies of data items

Question sources

Computer Science Exit Exam · 2018 · Question 46

Question 41 | Q00969

Which type of fragmentation in a distributed database divides a relation into subrelations defined by subsets of its columns, normally repeating a key to support reconstruction?

A. Replication

B. Location

C. Horizontal

D. Vertical

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer D: Vertical

Vertical fragmentation projects column subsets into separate fragments. Repeating an appropriate key or row identifier allows a join to reconstruct the original rows without spurious combinations. Horizontal fragmentation instead selects subsets of rows, reconstructed by union. Replication creates copies rather than dividing the relation by attributes. Location transparency hides where data resides; it does not name the column-splitting design operation. The corrected category therefore supports the existing vertical-fragmentation answer.

Why each choice is right or wrong

A - Incorrect: Replication creates copies of data rather than splitting a relation into column subsets.

B - Incorrect: Location transparency hides placement details; it is not a fragmentation method.

C - Incorrect: Horizontal fragmentation partitions rows, not columns.

D - Correct: Correct: vertical fragmentation splits columns and normally repeats a reconstruction key.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB35

> Location transparency hides which site holds a named object.

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 116 | lines 11-28

ii. Replication transparency:
• It allows to store copies of a data at multiple sites as shown in the above diagram.
• This is done to minimize access time to the required data.
iii. Fragmentation transparency:
• Allows to fragment a relation horizontally (create a subset of rows of a relation) or vertically (create a subset of columns of a relation).
iv. Increased reliability and availability:
• Reliability refers to system live time, that is, system is running efficiently most of the time. Availability is the probability that the system is continuously available (usable or accessible) during a time interval.
• A distributed database system has multiple nodes (computers) and if one fails then others are available to do the job.
v. Improved performance:
• A distributed DBMS fragments the database to keep data closer to where it is needed most.
• This reduces data management (access and modification) time significantly.

Fundamental of Database Systems module | PDF page 118 | lines 3-21

• Horizontal fragmentation
• It is a horizontal subset of a relation which contain those of rows which satisfy selection conditions.
• Consider the Employee relation with selection condition (DNO = 5). All rows satisfy this condition will create a subset which will be a horizontal fragment of Employee relation.
• A selection condition may be composed of several conditions connected by AND or OR.
• Vertical fragmentation
• It is a subset of a relation which is created by a subset of columns. Thus a vertical fragment of a relation will contain values of selected columns.
- Consider the Employee relation. A vertical fragment of can be created by keeping the values of Name, Bdate, Sex, and Address.
- Because there is no condition for creating a vertical fragment, each fragment must include the primary key attribute of the parent relation Employee. In this way all vertical fragments of a relation are connected.
• Representation
• There three rules that must be followed during fragmentation
• Completeness - if a relation r is decomposed into fragments r1, r2... rn ,

Question sources

Computer Science Exit Exam · 2018 · Question 82

Question 42 | Q00993

In two-phase locking, which phase permits releasing existing locks but forbids acquiring new locks?

A. Expanding phase

B. Waiting phase

C. Shrinking phase

D. All of the above

Hint

Track the first lock release and the transaction-completion boundary.

Answer C: Shrinking phase

Two-phase locking divides lock use into a growing phase and a shrinking phase. During growing, a transaction may acquire or upgrade locks; once it releases a lock, it enters shrinking and cannot acquire any new lock. Waiting describes delay on a conflicting request, not a separate 2PL phase defined by releasing locks. Because the growing phase allows acquisition instead of release, the all-of-the-above alternative also fails.

Why each choice is right or wrong

A - Incorrect: Incorrect: the growing/expanding phase acquires locks and does not release them.

B - Incorrect: Incorrect: waiting on a conflicting request is a delay condition, not the formal 2PL release phase.

C - Correct: Correct: shrinking permits release while forbidding new acquisitions.

D - Incorrect: Incorrect: the growing phase does not meet the release/no-new-lock rule.

Primary note proof

Study-note section ADB12

> Basic 2PL has a growing phase that acquires/upgrades locks and a shrinking phase that releases/downgrades them. Once any lock is released, no new lock may be acquired. The lock point is the instant the final lock is obtained. 2PL guarantees conflict serializability for well-formed transactions, but can deadlock and basic 2PL alone need not prevent cascading aborts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 86 | lines 16-24

> ◦ Locking (Growing) Phase:
> • A transaction applies locks (read or write) on desired data items one at a time.
> • acquires all locks but cannot release any locks.

◦ Unlocking (Shrinking) Phase:
• A transaction unlocks its locked data items one at a time.
• Releases locks but cannot acquire any new locks.
● Requirement:
◦ For a transaction these two phases must be mutually exclusively, that is, during

Question sources

National Exit Exam practice questions · Question 7

Computer Science Exit Exam practice collection · Question 65

Question 43 | Q01002

Which one of the SQL keywords is used to provide access?

A. REVOKE

B. DISTINCT

C. GRANT

D. None of the above

Hint

Track whether a privilege is being added or withdrawn and whether it applies to rows or columns.

Answer C: GRANT

GRANT gives specified users privileges on an object. REVOKE removes privileges.

Why each choice is right or wrong

A - Incorrect: REVOKE withdraws access privileges. (PDF pp. 54, 55.)

B - Incorrect: DISTINCT is not the module's access-authorization command. (PDF pp. 54, 55.)

C - Correct: GRANT supplies the privileges on tables or views. (PDF pp. 54, 55.)

D - Incorrect: GRANT is listed, so None is false. (PDF pp. 54, 55.)

Primary note proof

Study-note section ADB25

GRANT SELECT ON Catalogue TO Mohammed; permits reads. REVOKE DELETE, UPDATE (Price,Year) ON Catalogue FROM Mohammed; removes the specified table deletion and column update grants in SQL dialects supporting that form. DELETE applies to rows/table access rather than particular columns; UPDATE can be column-scoped. REVOKE removes permission, whereas GRANT gives it.

Study-note section ADB25

WITH GRANT OPTION permits a recipient to grant a privilege onward; removing an upstream grant can cascade or be restricted according to the chosen command and DBMS.

Study-note section ADB25

Roles group privileges.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 54 | lines 12-20

GRANT <privileges>
ON <object>
TO <users>
[WITH GRANT OPTION]
• <privileges> is a list of
SELECT <columns>, INSERT <columns>, DELETE, and UPDATE <columns>, or
simply ALL
• <users> is a list of user names or PUBLIC
• <object> is the name of a table or view

Fundamental of Database Systems module | PDF page 55 | lines 8-12

REVOKE <privileges>
ON <object>
FROM <users>
- If a user has the same privilege from other users then they keep it
- All privileges dependent on the revoked one are also revoked

Question sources

National Exit Exam practice questions · Question 16

Computer Science Exit Exam practice collection · Question 48

Question 44 | Q01156

is a tree data structure that corresponds to a relational algebra expression.

A. a | Query graph x

B. i? Query tree WA

C. '@| Query optimizer x

D. | None of the above x

Hint

Separate query representation, strategy selection and execution.

Answer B: i? Query tree WA

A query tree corresponds to a relational algebra expression, with relations as leaves and operations as internal nodes.

Why each choice is right or wrong

A - Incorrect: Query graph corresponds to relational calculus. (PDF pp. 34.)

B - Correct: Query tree is the exact defined algebraic tree. (PDF pp. 34.)

C - Incorrect: The optimizer chooses an execution strategy; it is not this data structure. (PDF pp. 34.)

D - Incorrect: The matching tree is explicitly listed. (PDF pp. 34.)

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB-S01

> A query tree corresponds to a relational-algebra expression: base relations appear at leaves and operators at internal nodes.

Study-note section ADB17

> A query tree has base relations at leaves and operators at internal nodes; a query graph emphasizes relations and predicates.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 34 | lines 5-7

> 2. Apply heuristics rules to optimize the internal representation.
> 3. A query execution plan is generated to execute groups of operations based on the access paths available on the files involved in the query.

Question sources

Computer Science Exit Exam practice questions · Question 17

Question 45 | Q01184

Which listed approaches use both undo and redo? Interpret Immediate Update as the conventional steal/no-force logged model and Deferred Update as no-steal/no-force; use the course's ARIES definition. Select all that apply.

A. Immediate Update

B. Deferred Update

C. ARIES Algorithm

D. None

Hint

Determine whether uncommitted changes or unflushed committed changes may exist on persistent pages.

Answer A, C: A. Immediate Update; C. ARIES Algorithm

Steal allows uncommitted changes to reach persistent pages, creating a possible need for undo. No-force allows committed changes to remain buffered, creating a possible need for redo. Immediate update under the specified combined policy therefore needs both. ARIES also performs redo and then undoes losers. The specified deferred no-steal/no-force model needs redo but no undo, so it does not qualify.

Why each choice is right or wrong

A - Correct: The stated steal/no-force policy requires undo and redo.

B - Incorrect: The specified deferred policy prevents persistent uncommitted effects and needs redo only.

C - Correct: ARIES repeats history with redo and reverses losers with undo.

D - Incorrect: Two listed approaches use both actions.

Primary note proof

Study-note section ADB28

> Steal permits a page with uncommitted changes to be written to disk; hence recovery may need undo. No-steal prevents those uncommitted effects reaching the persistent database. Force writes a transaction's updated data pages at commit; no-force permits committed dirty pages to remain buffered, hence recovery may need redo.

Study-note section ADB28

> For the conventional page/log model: steal/no-force needs undo and redo; steal/force needs undo but no redo; no-steal/no-force needs redo but no undo; no-steal/force needs neither data-page undo nor redo after a system crash.

Study-note section ADB29

> Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO.

Related reading (context, not primary note proof)

Study-note section ADB31

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 105 | lines 25-36

Check pointing
• Log file is used to recover failed DB but we may not know how far back in the log to search. Thus
• Checkpoint is a Point of synchronization between database and log file.
• Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.
• When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Fundamental of Database Systems module | PDF page 109 | lines 13-22

iii. Shadow Paging
• Maintain two page tables during life of a transaction: current page and shadow page table.
• When transaction starts, two pages are the same.
• Shadow page table is never changed thereafter and is used to restore database in event of failure.
• During transaction, current page table records all updates to database.
• When transaction completes, current page table becomes shadow page table.
X and Y: Shadow copies of data items
X' and Y': Current copies of data items

Fundamental of Database Systems module | PDF page 110 | lines 14-30

◦ Deferred Update:
• All transaction updates are recorded in the local workspace (cache)
• All modified data items in the cache is then written after a transaction ends its execution or after a fixed number of transactions have completed their execution.
• During commit the updates are first recorded on the log and then on the database
• If a transaction fails before reaching its commit point undo is not needed because it didn‘t change the database yet
• If a transaction fails after commit (writing on the log) but before finishing saving to the data base redoing is needed from the log
◦ Immediate Update:
• As soon as a data item is modified in cache, the disk copy is updated.
• These update are first recorded on the log and on the disk by force writing,

before the database is updated
• If a transaction fails after recording some change to the database but before reaching its commit point , this will be rolled back

Question sources

Computer Science model exam 2 · Question 5

Question 46 | Q01209

Which recovery mechanism preserves the committed page map while writing modified pages elsewhere, then atomically installs a new map at commit as an alternative to conventional log-based recovery?

A. Disk recovery

B. Shadow paging

C. Deferred update

D. Crash recovery

Hint

Separate update timing from update location and follow the commit boundary.

Answer B: Shadow paging

Shadow paging preserves the committed page map and writes modified pages to separate locations. Once the required pages are durable, commit atomically installs the new map; before that point, recovery can use the old committed map. Deferred update instead describes postponing persistent database changes until commit and does not itself specify this page-map mechanism. Disk recovery and crash recovery name recovery domains or events, not the particular algorithm requested.

Why each choice is right or wrong

A - Incorrect: Incorrect: disk recovery describes a recovery domain, not the specified preserved-page-map mechanism.

B - Correct: Correct: shadow paging preserves the committed map and installs the new map atomically at commit.

C - Incorrect: Incorrect: deferred update specifies commit timing, not this copy-on-write page-map algorithm.

D - Incorrect: Incorrect: crash recovery describes recovery after an event rather than the particular page-map mechanism.

Primary note proof

Study-note section ADB29

In-place update overwrites the original location. Shadow/copy-on-write update writes a new location while preserving the committed version. Shadow paging keeps an unchanged committed page map and a modified map; commit durably installs a new root/map pointer atomically after required pages are durable. Before commit, a crash uses the old map. It is an alternative to conventional log-based recovery, with allocation, fragmentation and concurrency costs; it is not primarily a special action done only during recovery.

Study-note section ADB29

Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO.

Related reading (context, not primary note proof)

Study-note section ADB26

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 102 | lines 25-28

- We study two approaches:
- log-based recovery, and
- shadow-paging
- We assume (initially) that transactions run serially, that is, one after the other.

Fundamental of Database Systems module | PDF page 109 | lines 13-22

iii. Shadow Paging
• Maintain two page tables during life of a transaction: current page and shadow page table.
• When transaction starts, two pages are the same.
• Shadow page table is never changed thereafter and is used to restore database in event of failure.
• During transaction, current page table records all updates to database.
• When transaction completes, current page table becomes shadow page table.
X and Y: Shadow copies of data items
X' and Y': Current copies of data items

Question sources

Computer Science model exam 2 · Question 30

Question 47 | Q01225

Which specific DBMS component directly handles undo of incomplete persistent changes after failure to support transaction atomicity? Choose the specialized component rather than its broader parent subsystem.

A. Transaction management component of DBMS

B. Application Programmer

C. Concurrency control component of DBMS

D. Recovery management component of DBMS

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer D: D. Recovery management component of DBMS

Atomicity requires all transaction effects or none. After failure, recovery can undo incomplete persistent changes to enforce that outcome. Transaction management is the broader subsystem containing these responsibilities, but the question asks for the specialized component. Concurrency control primarily addresses isolation, and application code alone cannot provide the DBMS's crash-recovery undo of partially persisted work.

Why each choice is right or wrong

A - Incorrect: This is the parent subsystem, not the specialized failure-undo component requested.

B - Incorrect: Application logic alone does not provide durable crash undo.

C - Incorrect: Concurrency control primarily enforces isolation.

D - Correct: Recovery directly supports atomicity by undoing incomplete effects.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward.

Study-note section ADB27

> Write-ahead logging requires the update log needed for recovery to reach stable storage before the corresponding changed data page reaches disk.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 66 | lines 4-13

- Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
- Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.
- Isolation: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary
- Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.

Example

Fundamental of Database Systems module | PDF page 97 | lines 4-13

i. Purpose of Database Recovery
◦ To bring the database into the last consistent state, which existed prior to the failure.
◦ To preserve transaction properties (Atomicity &and Durability).
• The recovery manager of a DBMS is responsible to ensure
• atomicity by undoing the action of transaction that do not commit
• Durability by making sure that all actions of committed transaction survive system crash
ii. Types of Failure
◦ The database may become unavailable for use due to :

Fundamental of Database Systems module | PDF page 66 | lines 22-31

- Atomicity- either all or none of the above operation will be done - this is materialized by transaction management component of DBMS
- Consistency-the sum of CA2090 and SB2359 be unchanged by the execution of Ti i.e 8500- this is the responsibility of application programmer who codes the transaction
- Isolation- when several transaction are being processed concurrently on a data item they may create many inconsistent problems. So handling such case is the responsibility of Concurrency control component of the DBMS
- Durability - once Ti writes its update this will remain there when the database restarted from failure . This is the responsibility of recovery management components of the DBMS

Question sources

Computer Science model exam 2 · Question 46

Question 48 | Q01238

If a database server is referenced in a distributed transaction, the value of its commit point strength determines which role it plays in the _____.

A. Two-phase commit

B. Transaction locking

C. Two-phase locking

D. Checkpoints

Hint

Distinguish coordinated transaction outcome from concurrent access and local storage.

Answer A: Two-phase commit.

ADB39 explicitly links Oracle COMMIT_POINT_STRENGTH to choosing the commit point site in distributed transactions. This is a role in two-phase commit, which coordinates an atomic outcome among participants. It is not a setting for acquiring or releasing locks, nor for checkpoints that bound recovery work. Higher eligible strength is preferred for the commit point.

Why each choice is right or wrong

A - Correct: Commit-point strength selects an Oracle distributed-commit role within the two-phase commit mechanism.

B - Incorrect: Transaction locking controls concurrent access rather than choosing the distributed commit point site.

C - Incorrect: Two-phase locking governs lock acquisition and release, not the distributed commit decision.

D - Incorrect: Checkpoints record recovery state and do not choose an Oracle commit point site.

Primary note proof

Study-note section ADB39

> Two-phase commit (2PC) first asks participants to prepare and durably promise they can commit. If all vote yes, the coordinator records commit durably and tells participants; otherwise it decides abort. A prepared participant must retain enough state to obey the final decision. Coordinator/communication failure can leave it in doubt and block progress until the decision is learned. Timeout after preparation is not permission to invent a commit or abort.

Study-note section ADB39

> Oracle's COMMIT_POINT_STRENGTH helps choose the commit point site in its distributed transaction mechanism, with higher eligible strength preferred.

Study-note section ADB12

> Two-phase locking orders access to data; two-phase commit coordinates a distributed commit decision.

Related reading (context, not primary note proof)

Study-note section ADB30

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 123 | lines 15-29

• Distributed commit:
• A transaction may be fragmented and they may be executed by a number of sites. This require a two or three-phase commit approach for transaction commit.
• Distributed deadlock:
• Since transactions are processed at multiple sites, two or more sites may get involved in deadlock. This must be resolved in a distributed manner.
6.5.1 Distributed Concurrency control
i. Primary site technique: A single site is designated as a primary site which serves as a coordinator for transaction management.
• Transaction management:
- Concurrency control and commit are managed by this site.
- In two phase locking, this site manages locking and releasing data items. If all transactions follow two-phase policy at all sites, then serializability is guaranteed.
- Advantages:

Question sources

Computer Science model exam 2 · Question 59

Question 49 | Q01240

What is the distinctive purpose of query optimization within the broader query-processing pipeline? Select the most specific description.

A. Both are used to process SQL statements to return answers.

B. Both filter retrieved rows by a date/time condition.

C. Both improve the performance of the database by efficiently executing a query.

D. Both are important in executing queries to access information from the DB.

Hint

Separate query representation, strategy selection and execution.

Answer C: C. Both improve the performance of the database by efficiently executing a query.

Query optimization selects a suitable efficient execution strategy among equivalent ways of answering a query. C explicitly states that efficiency objective. Returning answers and accessing information are broader descriptions of processing and do not distinguish optimization's particular purpose. Filtering data by a date or time is a query condition rather than the general goal of choosing a plan.

Why each choice is right or wrong

A - Incorrect: Returning answers describes the broad pipeline but omits the distinctive efficiency objective.

B - Incorrect: Temporal filtering is query logic, not the general purpose of optimization.

C - Correct: Efficient execution is the specific objective of selecting a suitable plan.

D - Incorrect: Accessing information describes the broad pipeline rather than optimization's efficiency role.

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB17

> Query processing is the whole activity; selecting a suitable efficient strategy is optimization.

Study-note section ADB17

> DDL is not a mandatory stage before each SELECT, so authentication→DDL→DML permutations do not describe the processing sequence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 27 | lines 4-15

> What is Query Processing?
> Steps required to transform high level SQL query into a correct and —efficient‖ strategy for execution and retrieval.
> What is Query Optimization?
> ✓ The activity of choosing a single —efficient‖ execution strategy (from hundreds) as determined by database catalog statistics.
> ✓ The aim is to answer which relational algebra expression, equivalent to the given query, will lead to the most efficient solution plan?
> ✓ For each algebraic operator, what algorithm (of several available) do we use to

compute that operator?
✓ How do operations pass data (main memory buffer, disk buffer,...)?
✓ Will this plan minimize resource usage? (CPU/Response Time/Disk)

Fundamental of Database Systems module | PDF page 30 | lines 17-30

1.8 Query Optimization
• Everyone wants the performance of their database to be optimal. In particular, there is often a requirement for a specific query or object that is query based, to run faster.
• Problem of query optimization is to find the sequence of steps that produces the answer to user request in the most efficient manner, given the database structure.
• The performance of a query is affected by the tables or views that underlies the query and by the complexity of the query.
• Given a request for data manipulation or retrieval, an optimizer will choose an optimal plan for evaluating the request from among the manifold alternative strategies. i.e. there are many ways (access paths) for accessing desired file/record.
• Hence ,DBMS is responsible to pick the best execution strategy based on various considerations(Least amount of I/O and CPU resources.)
• Example: Consider relations r(AB) and s(CD). We require r X s.
• Method 1 :

Question sources

Computer Science model exam 2 · Question 61

Question 50 | Q01245

What is the objective of a Serializable schedule?

A. Find a Non-serial schedule that allows the interleaving of a transaction and produces the same result as a serial schedule.

B. Find a Serial schedule that allows the interleaving of a transaction and produces the same result as a serial schedule.

C. Find a Serial schedule that doesn't allow the interleaving of a transaction and produces the same result as a serial schedule.

D. Find a Non-serial schedule that doesn't allow the interleaving of a transaction and produces the same result as a serial schedule.

Hint

Distinguish transaction ordering, conflict conditions and full execution equivalence.

Answer A: Find a Non-serial schedule that allows the interleaving of a transaction and produces the same result as a serial schedule.

Serial schedules run each transaction consecutively. Serializability permits nonserial interleaving while preserving the effect of a complete serial schedule (p.72). A expresses this concurrency objective.

Why each choice is right or wrong

A - Correct: Interleaving plus serial-equivalent effect meets the objective.

B - Incorrect: A serial schedule cannot interleave transactions as claimed.

C - Incorrect: This describes serial execution, not the benefit of serializable interleaving.

D - Incorrect: Nonserial execution is distinguished by departure from consecutive transaction execution; this denies the intended interleaving.

Primary note proof

Study-note section ADB09

A serial schedule executes each transaction's operations consecutively. A serializable schedule is equivalent to some serial order; it may interleave transactions. Equality of one final value is insufficient evidence because different executions can accidentally coincide. Two operations conflict when they belong to different transactions, access the same item and at least one writes. Read/read pairs do not conflict.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 72 | lines 6-18

- Serial schedule:
  - A schedule S is serial if, for every transaction Ti participating in the schedule, all the operations of Ti are executed consecutively in the schedule. Otherwise, the schedule is called nonserial schedule.
  - For example, in the banking example suppose there are two transaction where one transaction calculate the interest on the account and another deposit some money into the account. hence the order of execution is important for the final result.
- Serializable schedule:
  - a schedule whose effect on any consistent database instance is identical to that of some complete serial schedule over the set of committed transactions in S.
  - A nonserial schedule S is serializable , is equivalent to say that it is correct to the result of one of the serial schedule .Example,

Question sources

Computer Science model exam 2 · Question 66

Question 51 | Q01249

Under the course data-model type-constructor convention, which statements are false? Select all that apply. Interpret type constructor as a data-model term, not an executable Java/C++ object constructor, and consider OID over one object's lifetime.

A. Object Identifier is immutable.

B. Type constructor is used to create a new object.

C. Tuple is a collection of records.

D. The atom constructor is a multi-value constructor.

Hint

Classify the shape of a value separately from an operation that creates an instance.

Answer B, C, D: B. Type constructor is used to create a new object.; C. Tuple is a collection of records.; D. The atom constructor is a multi-value constructor.

An OID remains stable over the object's lifetime, so A is true. A data-model type constructor specifies the shape of a value rather than executing creation of a new object. A tuple groups named components into one structured record, while an atom is one basic value. B, C and D therefore state incorrect properties under the specified convention.

Why each choice is right or wrong

A - Incorrect: OID stability over the object lifetime is true, so it does not answer the negative stem.

B - Correct: This is false under the note's convention: a type constructor specifies shape rather than executing object creation.

C - Correct: A tuple is one structured record of components, not a collection of records.

D - Correct: An atom represents one basic atomic value rather than a multivalued collection.

Primary note proof

Study-note section ADB04

> In a common object model, an object is described by (OID,type constructor,state). An atom is one basic value, such as an integer, real, Boolean or string. A tuple constructor groups named component values into one structured record, such as (Street:'A',City:'B'). It is not a collection of records.

Study-note section ADB04

> A type constructor defines the shape of a value; it is not the same as an executable constructor operation that creates a new object.

Study-note section ADB01

> The object identifier (OID) distinguishes objects independently of their attribute values and remains stable over that object's lifetime.

Question sources

Computer Science model exam 2 · Question 70

Question 52 | Q01264

Assume transaction A holds a shared lock R. If transaction B also requests for a shared lock on R

A. It will result in a deadlock situation

B. It will be granted as soon as it is released by A

C. It will immediately be granted

D. It will immediately be rejected

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer C: It will immediately be granted

The shared-mode rule permits more than one transaction to hold a read lock on X, while excluding write locks (p.82). With only A's shared lock specified, B's shared request is compatible.

Why each choice is right or wrong

A - Incorrect: Two compatible shared requests do not produce the stated deadlock.

B - Incorrect: Waiting for A is unnecessary for compatible reads.

C - Correct: The shared request is compatible and may be granted immediately under the stated conditions.

D - Incorrect: The shared rule supplies no rejection condition here.

Primary note proof

Study-note section ADB11

> A shared (S) lock permits reading; multiple S locks on the same item can coexist. An exclusive (X) lock permits its holder to read and write and conflicts with another S or X on that item. Locks on different items do not conflict merely because one is exclusive. These are the basic lock-based rules; multiversion systems can permit additional snapshot-reader behavior.

Study-note section ADB11

> One transaction may request a lock on an already locked item: it is granted if compatible, otherwise waits or aborts under the policy.

Study-note section ADB11

> The lock manager decides requests; the lock table stores granted locks and waiting requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 82 | lines 20-26

• Shared mode: shared lock (X)
• More than one transaction can apply share lock on X for reading its value but no write lock can be applied on X by any other transaction.
• Exclusive mode: Write lock (X)
• Only one write lock on X can exist at any time and no shared lock can be applied by any other transaction on X.
• Conflict matrix

Fundamental of Database Systems module | PDF page 83 | lines 23-25

• Reads cannot conflict, so more than one transaction can hold shared locks simultaneously on same item.
• Exclusive lock gives transaction exclusive access to that item.

Question sources

Computer Science model exam 2 · Question 85

Question 53 | Q01266

In the dirty read problem, one transaction reads:

A. an uncommitted value of another transaction

B. the committed value for another transaction

C. another transaction

D. one transaction commits another transaction

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer A: an uncommitted value of another transaction

The temporary-update/dirty-read definition involves another transaction accessing an updated item before the failed writer's value is restored. The hazard is observing the other transaction's uncommitted value.

Why each choice is right or wrong

A - Correct: This identifies the unsafe value from the other transaction.

B - Incorrect: A committed value lacks the defining uncommitted-update hazard.

C - Incorrect: Transactions read data items; this does not identify the dirty value.

D - Incorrect: One transaction committing another is not the read anomaly described.

Primary note proof

Study-note section ADB08

Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 70 | lines 2-12

ii. The Temporary Update (or Dirty Read) Problem
◦ This occurs when one transaction updates a database item and then the transaction fails for some reason .
◦ The updated item is accessed by another transaction before it is changed back to its original value. Based on the above scenario:
iii. The Incorrect Summary Problem
◦ If one transaction is calculating an aggregate summary function on a number of records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.
Example: T1 would like to add the values of A=10, B=20 and C=30. after the values are read by

Question sources

Computer Science model exam 2 · Question 87

Question 54 | Q01381

Which sequence matches the course's logical query-processing pipeline for a SELECT, from checking the request to returning its result?

A. Parsing and semantic analysis → logical translation → query optimization → evaluation → output

B. DDL->DML->query optimizer-> Authentication->output

C. DML->query optimizer-> Authentication-> DDL-> output

D. All of the mentioned

Hint

Separate query representation, strategy selection and execution.

Answer A: A. Parsing and semantic analysis → logical translation → query optimization → evaluation → output

The request is first checked for grammar and meaning, then translated into a logical representation. Optimization selects equivalent operations and physical access methods; evaluation executes that plan and returns the output. Option A follows those roles in order. The other sequences treat DDL and DML language categories as processing stages and place unrelated checks after optimization, so they do not fit.

Why each choice is right or wrong

A - Correct: This sequence follows checking, representation, strategy selection and execution.

B - Incorrect: DDL/DML categories do not form these phases, and checking is misplaced.

C - Incorrect: DDL is not a mandatory post-optimization phase of SELECT processing.

D - Incorrect: Only option A follows the pipeline, so all sequences are not correct.

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB17

> DDL is not a mandatory stage before each SELECT, so authentication→DDL→DML permutations do not describe the processing sequence.

Study-note section ADB17

> Query processing is the whole activity; selecting a suitable efficient strategy is optimization.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 29 | lines 9-20

> 1.6 Query Processing Steps
> • Processing can be divided into 4 :Decomposition, Optimization , Execution and Code generation
> 1.7 Query Decomposition
> ● It is the process of transforming a high level query into a relational algebra query, and to check that the query is syntactically and semantically correct. It Consists of parsing and validation
> Typical stages in query decomposition are:
> i. Analysis: lexical and syntactical analysis of the query (correctness) based on attributes, data type.. ,. Query tree will be built for the query containing leaf node for base relations, one or many non-leaf nodes for relations produced by relational algebra operations and root node for the result of the query. Sequence of operation is from the leaves to the

Fundamental of Database Systems module | PDF page 30 | lines 17-30

> 1.8 Query Optimization
> • Everyone wants the performance of their database to be optimal. In particular, there is often a requirement for a specific query or object that is query based, to run faster.
> • Problem of query optimization is to find the sequence of steps that produces the answer to user request in the most efficient manner, given the database structure.

• The performance of a query is affected by the tables or views that underlies the query and by the complexity of the query.
• Given a request for data manipulation or retrieval, an optimizer will choose an optimal plan for evaluating the request from among the manifold alternative strategies. i.e. there are many ways (access paths) for accessing desired file/record.
• Hence ,DBMS is responsible to pick the best execution strategy based on various considerations(Least amount of I/O and CPU resources.)
• Example: Consider relations r(AB) and s(CD). We require r X s.
• Method 1 :

Question sources

Database Fundamentals practice questions · Question 56

Question 55 | Q01501

In object-oriented database, the keyword "inverse" is used in which of the following?

A. Relationship

B. All choices are correct

C. Class

D. Attribute

Hint

Distinguish schema definitions, queries and opposite navigation directions.

Answer A: Relationship.

An inverse declaration connects two navigation directions of the same association, such as Department.employees and Employee.department. ADB05 explains that it maintains their relationship consistency. It does not declare a class, establish inheritance or name an ordinary scalar attribute. Thus Relationship is the applicable category, and the inclusive option is false.

Why each choice is right or wrong

A - Correct: Inverse pairs the two directions of one relationship and expresses their consistency.

B - Incorrect: The other categories are not all valid uses of the inverse declaration.

C - Incorrect: Inverse is not the keyword that declares a class or establishes a class hierarchy.

D - Incorrect: The declaration links relationship directions rather than declaring an ordinary scalar attribute.

Primary note proof

Study-note section ADB05

> An inverse declaration pairs opposite directions of the same relationship. Conceptually, Department.employees is inverse to Employee.department: when an employee belongs to department D, the corresponding employee appears in D's collection. The declaration expresses relationship consistency; it is not an inheritance keyword or an ordinary scalar attribute. An MCQ calling ODL both a DDL and a language for developing logical schemas offers overlapping true descriptions, so its exact wording matters.

Question sources

Mock exam 1 · Question 17

Question 56 | Q01621

Which statements describe ODL? Select all that apply. In A, 'implement' specifically means choosing or carrying out physical storage implementation, rather than expressing a logical definition.

A. A method to implement a logical schema

B. All choices are correct

C. Used to develop logical schemas

D. A data definition language for OODB

Hint

Distinguish schema definitions, queries and opposite navigation directions.

Answer C, D: C. Used to develop logical schemas; D. A data definition language for OODB

ODL is the schema-definition language for object databases and describes classes or interfaces, attributes, operations and relationships. It is therefore both a data definition language and a language used to develop logical schemas. The question defines implementation in A as physical storage work, which is a different task. Because A is false in that sense, All choices are correct is also false.

Why each choice is right or wrong

A - Incorrect: Physical storage implementation is distinct from ODL logical definition.

B - Incorrect: A is false under the specified implementation meaning.

C - Correct: ODL develops logical object schemas.

D - Correct: ODL is a data definition language for object databases.

Primary note proof

Study-note section ADB05

> Object Definition Language (ODL) is the ODMG schema-definition language for object databases; it describes interfaces/classes, attributes, operations and relationships. Object Query Language (OQL) expresses queries. Host-language bindings connect these definitions and operations to languages such as Java or C++. Defining a logical schema is different from implementing its storage.

Question sources

Mock exam 2 · Question 10

Question 57 | Q01622

What are ACID properties of Transactions?

A. Automatically, Concurrency, Isolation, Durability

B. Atomicity, Consistency, Inconsistent, Durability

C. Atomicity, Consistency, Isolation, Database

D. Atomicity, Consistency, Isolation, Durability

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer D: Atomicity, Consistency, Isolation, Durability

Page 66 lists atomicity, consistency preservation, isolation, and durability/permanency. Only D preserves all four terms.

Why each choice is right or wrong

A - Incorrect: Automatically and concurrency replace atomicity and consistency.

B - Incorrect: Inconsistent replaces isolation.

C - Incorrect: Database replaces durability.

D - Correct: All four names match the listed properties.

Primary note proof

Study-note section ADB06

A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 66 | lines 4-13

- Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
- Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.
- Isolation: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary
- Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.

Example

Question sources

Mock exam 2 · Question 11

Question 58 | Q01623

Which of the following commands is used to save any transaction permanently into the database?

A. Rollback

B. None of the choices are correct

C. Savepoint

D. Commit

Hint

Distinguish completing a transaction from marking an intermediate recovery position.

Answer D: Commit

COMMIT successfully completes the transaction and requests a durable committed outcome. ROLLBACK instead cancels uncommitted changes. SAVEPOINT marks an intermediate position to which later work can be rolled back, retaining earlier work within the same transaction; it neither completes that transaction nor makes the retained changes independently durable. Because COMMIT is offered, the none-of-the-choices alternative is false. D therefore identifies the permanent transaction-completion command.

Why each choice is right or wrong

A - Incorrect: Incorrect: ROLLBACK cancels uncommitted work rather than accepting it permanently.

B - Incorrect: Incorrect: COMMIT is offered and is the requested completion command.

C - Incorrect: Incorrect: SAVEPOINT marks an intermediate rollback position inside the still-open transaction; it does not commit.

D - Correct: Correct: COMMIT completes the transaction successfully and requests a durable outcome.

Primary note proof

Study-note section ADB-S02

> In SQL transaction-block notation, BEGIN starts a block, COMMIT completes it successfully, and ROLLBACK cancels its uncommitted changes. Committing requests a durable transaction outcome; it is distinct from merely finishing the last read or write operation.

Study-note section ADB-S02

> SAVEPOINT marks an intermediate position inside the current transaction. ROLLBACK TO discards changes made after that position while retaining earlier work in the transaction. A savepoint is not an independent durable commit, and rolling back to it does not itself commit the retained changes.

Related reading (context, not primary note proof)

Study-note section ADB06

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 65 | lines 4-7

◦ Committed state -ensures that all the changes done on a record by a transition were done persistently
◦ Failed state happens when a transaction is aborted during its active state or if one of the rechecking is fails

Fundamental of Database Systems module | PDF page 77 | lines 11-13

● BEGIN TRANSACTION statement and end with the COMMIT or ROLLBACK statement.
● SQL transaction Examples:

Fundamental of Database Systems module | PDF page 65 | lines 17-19

• [commit,T]: Records that transaction T has completed successfully, and affirms that its effect can be committed (recorded permanently) to the database.
• [abort,T]: Records that transaction T has been aborted.

Question sources

Mock exam 2 · Question 12

Question 59 | SM0003

Which one of the following is NOT correct about transaction failure?

A. Transactions fail if logical errors are detected

B. Catastrophes can be a cause of transaction failure

C. Concurrency control enforcement avoids transaction failure

D. Transactions fail if disk blocks lose their data

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer C: Concurrency control enforcement avoids transaction failure

The failure list includes logical programming errors, disk blocks losing data, and physical catastrophes. It also explicitly says concurrency enforcement may abort a transaction for serializability or deadlock (p.98). Thus claiming it avoids transaction failure is the incorrect statement.

Why each choice is right or wrong

A - Incorrect: Logical errors are explicitly a cause.

B - Incorrect: Physical problems and catastrophes are explicitly listed.

C - Correct: Concurrency enforcement itself may abort transactions, contradicting the absolute claim.

D - Incorrect: Loss of disk-block data is explicitly listed.

Primary note proof

Study-note section ADB26

Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 98 | lines 4-31

- What causes a Transaction to fail

1. A computer failure (system crash):
A hardware or software error occurs in the computer system during transaction execution. If the hardware crashes, the contents of the computer‘s internal memory may be lost.
2. A transaction or system error:
Some operation in the transaction may cause it to fail, such as integer overflow or division by zero. Transaction failure may also occur because of erroneous parameter values or because of a logical programming error. In addition, the user may interrupt the transaction during its execution.
3. Exception conditions detected by the transaction:
◦ Certain conditions forces cancellation of the transaction.
◦ For example, data for the transaction may not be found. such as insufficient account balance in a banking database, may cause a transaction, such as a fund withdrawal from that account, to be canceled.
4. Concurrency control enforcement:
The concurrency control method may decide to abort the transaction, to be restarted later, because it violates serializability or because several transactions are in a state of deadlock (see Chapter 2).
5. Disk failure:
Some disk blocks may lose their data because of a read or write malfunction or because of a disk read/write head crash. This may happen during a read or a write operation of the transaction.
6. Physical problems and catastrophes:
This refers to an endless list of problems that includes power or air-conditioning failure, fire, theft, overwriting disks or tapes by mistake
Transaction Manager : Accepts transaction commands from an application, which tell the transaction manager when transactions begin and end, as well as information about the

expectations of the application.
The transaction processor performs the following tasks:

Question sources

MOE Computer Science Exit Exam · Question 1-3

National Computer Science Exit Exam · Question 10

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 59

Question 60 | SM0004

Which one of the data updates best describes: 'All modified data items in the cache is written either after transaction ends its execution or after a pre-determined number of transactions have concluded their
execution'?

A. Immediate update

B. In-place update

C. Shadow update

D. Deferred update

Hint

Separate update timing from update location and follow the commit boundary.

Answer D: Deferred update

The deferred-update passage on p.110 almost exactly reproduces the stem: modified cache items are written after a transaction ends or a fixed number finish.

Why each choice is right or wrong

A - Incorrect: Immediate update writes as soon as an item changes in cache.

B - Incorrect: In-place describes overwriting the disk version, not the delay policy.

C - Incorrect: Shadow update writes a separate disk location.

D - Correct: Deferred update has precisely the stated timing.

Primary note proof

Study-note section ADB29

Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO. Redo-only and NO-UNDO/REDO mean the same thing, so those cannot be separate unique answers. Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Study-note section ADB29

Shadow/copy-on-write update writes a new location while preserving the committed version.

Study-note section ADB29

In-place update overwrites the original location.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 110 | lines 14-30

◦ Deferred Update:
• All transaction updates are recorded in the local workspace (cache)
• All modified data items in the cache is then written after a transaction ends its execution or after a fixed number of transactions have completed their execution.
• During commit the updates are first recorded on the log and then on the database
• If a transaction fails before reaching its commit point undo is not needed because it didn‘t change the database yet
• If a transaction fails after commit (writing on the log) but before finishing saving to the data base redoing is needed from the log
◦ Immediate Update:
• As soon as a data item is modified in cache, the disk copy is updated.
• These update are first recorded on the log and on the disk by force writing, before the database is updated
• If a transaction fails after recording some change to the database but before reaching its commit point , this will be rolled back

Fundamental of Database Systems module | PDF page 111 | lines 2-9

◦ Shadow update:
• The modified version of a data item does not overwrite its disk copy but is written at a separate disk location.
• Multiple version of the same data item can be maintained
• Thus the old value (before image BFIM) and the new value (AFIM) are kept in the disk
• No need of Log for recovery
◦ In-place update: The disk version of the data item is overwritten by the cache

Question sources

MOE Computer Science Exit Exam · Question 1-4

National Computer Science Exit Exam · Question 41

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 60

Question 61 | SM0015

T1 writes X but has not committed. T2 reads that value; T1 then fails and its write is rolled back. Which anomaly did T2 experience?

A. Nonrepeatable-read problem

B. It is dirty read

C. It is an incorrect summary problem

D. It is a lost update problem

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer B: B. It is dirty read

T2 consumes a value written by another transaction before that transaction commits. The writer's later failure makes this a dirty-read dependency requiring reconsideration of dependent work. A nonrepeatable read involves repeated reads seeing a committed change, while an incorrect summary mixes inconsistent versions in an aggregate. Lost update involves a stale overwrite, which this trace does not describe.

Why each choice is right or wrong

A - Incorrect: No repeated read observing a committed change is described.

B - Correct: T2 reads another transaction's uncommitted write.

C - Incorrect: No aggregate combines inconsistent versions.

D - Incorrect: No stale value is written over another update.

Primary note proof

Study-note section ADB08

> Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 70 | lines 2-12

> ii. The Temporary Update (or Dirty Read) Problem
> ◦ This occurs when one transaction updates a database item and then the transaction fails for some reason .
> ◦ The updated item is accessed by another transaction before it is changed back to its original value. Based on the above scenario:

iii. The Incorrect Summary Problem
◦ If one transaction is calculating an aggregate summary function on a number of records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.
Example: T1 would like to add the values of A=10, B=20 and C=30. after the values are read by

Question sources

MOE Computer Science Exit Exam · Question 4-3

National Computer Science Exit Exam · Question 58

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 61

Question 62 | SM0045

When two transactions that access the same database items have their operations interleaved in a way that makes the value of some database item incorrect. Which option correctly identifies this?

A. It is a temporary read problem

B. It is a dirty read problem

C. It is an incorrect summary problem

D. It is a lost update problem

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer D: It is a lost update problem

The lost-update definition on p.68 is the wording in the stem. The other anomalies concern reading a temporary failed update or mixing summary values.

Why each choice is right or wrong

A - Incorrect: Temporary-update/dirty-read requires access to an uncommitted failed update, not merely the interleaving described.

B - Incorrect: Dirty read is the temporary-update anomaly.

C - Incorrect: Incorrect summary concerns aggregate evaluation over mixed versions.

D - Correct: Lost update matches the stated definition.

Primary note proof

Study-note section ADB08

Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 68 | lines 4-7

i. The Lost Update Problem
◦ This occurs when two transactions that access the same database items have their operations interleaved in a way that makes the value of some database item incorrect.

Fundamental of Database Systems module | PDF page 70 | lines 2-12

ii. The Temporary Update (or Dirty Read) Problem
◦ This occurs when one transaction updates a database item and then the transaction fails for some reason .
◦ The updated item is accessed by another transaction before it is changed back to its original value. Based on the above scenario:
iii. The Incorrect Summary Problem
◦ If one transaction is calculating an aggregate summary function on a number of records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.
Example: T1 would like to add the values of A=10, B=20 and C=30. after the values are read by

Question sources

MOE Computer Science Exit Exam · Question 13-3

National Computer Science Exit Exam · Question 40

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 62

Question 63 | SM0046

Which listed policy prevents lock-wait deadlock without comparing transaction-age timestamps? Here 'wait-wait' means allowing every conflicting request to wait with no ordering or cycle restriction.

A. Wait-wait

B. No-wait

C. Wound-wait

D. Wait-die

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer B: No-wait

No-wait aborts immediately when a lock request conflicts, so transactions cannot form a lock-wait cycle and no age comparison is needed. Wait-die and wound-wait use relative transaction ages to decide who waits or aborts. Under the explicitly defined wait-wait alternative, every conflicting request may wait without a cycle-prevention restriction, so a transaction can hold one item while waiting for another in a cycle. That alternative therefore fails the deadlock-prevention requirement.

Why each choice is right or wrong

A - Incorrect: Incorrect: unrestricted waiting permits a hold-and-wait cycle, so this defined policy does not prevent deadlock.

B - Correct: Correct: immediate abort on a conflicting request prevents lock waits without comparing ages.

C - Incorrect: Incorrect: wound-wait compares requester and holder ages to choose waiting or aborting a younger holder.

D - Incorrect: Incorrect: wait-die compares ages to choose whether the requester waits or aborts.

Primary note proof

Study-note section ADB13

> Fixed resource ordering and conservative 2PL prevent cycles. A no-wait policy aborts immediately on an incompatible lock request and needs no age timestamp. A timeout aborts long waits but may mistake slow progress for deadlock. Starvation is indefinite postponement while other work proceeds, without a required cycle. Fair queues, sensible victim selection and retaining original transaction timestamps on retry reduce starvation; deadlock freedom alone does not prove fairness.

Study-note section ADB14

> A smaller timestamp means an older transaction. In wait-die, an older requester may wait for a younger holder; a younger requester needing an older holder aborts itself. In wound-wait, an

older requester aborts the younger holder; a younger requester waits for the older holder. Always identify requester versus holder before applying the names.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 88 | lines 6-28

• Transaction Timestamp(TS(T))
− We can prevent deadlocks by giving each transaction a priority and ensuring that lower priority transactions are not allowed to wait for higher priority transactions (or vice versa).
− One way to assign priorities is to give each transaction a timestamp when it starts up.
− it is a unique identifier given to each transaction based on time in which it is started. i.e if T1 starts before T2 , TS(T1)<TS(T2)
− The lower the timestamp, the higher the transaction's priority, that is, the oldest transaction has the highest priority.
− If a transaction Ti requests a lock and transaction Tj holds a conflicting lock, the lock manager can use one of the following two policies: Wait-die & Wound-wait
● Wait-die
◦ If Ti has higher priority, it is allowed to wait; otherwise it is aborted.
◦ An older transaction is allowed to wait on a younger transaction.
◦ A younger transaction requesting an item held by an older transaction is aborted
◦ If TS(Ti) < TS(Tj), then (Ti older than Tj)Ti is allowed to wait.
◦ Otherwise (Ti younger than Tj)Abort Ti (Ti dies) and restart it later with the same timestamp
T1(ts =10): M is locked , K is requested
T2 (ts =20): K is locked, Z is requested
88

Fundamental of Database Systems module | PDF page 89 | lines 4-13

• Wound-wait
- The opposite of wait-die
- If Ti has higher priority, abort Tj; otherwise Ti waits.
- A younger transaction is allowed to wait on an older one
- An older transaction requesting an item held by a younger transaction preempts the younger transaction by aborting it.
- If TS(Ti) < TS(Tj), then (Ti older than Tj),Abort Tj (Ti wounds Tj) and restart Tj later with the same timestamp
- Otherwise (Ti younger than Tj)Ti is allowed to wait
T2 (ts =20)

Question sources

MOE Computer Science Exit Exam · Question 14-1

National Computer Science Exit Exam · Question 100

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 63

Question 64 | SM0047

A database management system (DBMS) has _____ to control locks.

A. Lock manager

B. Lock table

C. Query optimizer

D. Query processor

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer A: Lock manager

Page 83 distinguishes the manager that manages locks from the table it uses to store lock identities and modes. The controlling component is the lock manager.

Why each choice is right or wrong

A - Correct: This component manages locks.

B - Incorrect: The table stores lock information for the manager.

C - Incorrect: Optimization selects execution strategies, not lock ownership.

D - Incorrect: The named lock-control component is the manager, not query processing.

Primary note proof

Study-note section ADB11

> A transaction must own the appropriate lock before access and can release only its own locks. S→X is an upgrade, possible only after conflicting other holders leave. X→S is a downgrade. The lock manager decides requests; the lock table stores granted locks and waiting requests. One transaction may request a lock on an already locked item: it is granted if compatible, otherwise waits or aborts under the policy. No double-locking is not a prohibition on shared readers.

Study-note section ADB17

> Query processing is the whole activity; selecting a suitable efficient strategy is optimization.

Study-note section ADB17

> A physical execution plan adds scan/index choices and algorithms.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 83 | lines 2-8

• Lock Manager:
• Managing locks on data
items.
• Lock table:
• Lock manager uses it to store the identify of transaction locking a data
item, the data item, lock mode and pointer to the next data item locked.
One simple way to implement a lock table is through linked list.

Question sources

MOE Computer Science Exit Exam · Question 14-2

National Computer Science Exit Exam · Question 75

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 64

Question 65 | SM0205

In a columnar database, data is stored:

A. Row-wise

B. In a hierarchical structure

C. Column-wise

D. In a network model

Hint

Distinguish operational transactions, analytical workloads and physical storage layout.

Answer C: Column-wise.

ADB41 defines columnar storage as grouping values by column, which avoids reading irrelevant columns and can improve compression for scans. Row-oriented storage instead groups each record's fields. Hierarchical and network models are logical representation alternatives rather than the physical column-oriented layout requested by the stem. Columnar layout alone does not determine the logical data model.

Why each choice is right or wrong

A - Incorrect: Row-wise grouping is row-oriented storage, the contrasting physical layout.

B - Incorrect: A hierarchy describes parent-child organization, not the columnar physical storage arrangement.

C - Correct: Columnar storage groups the values of each column together.

D - Incorrect: A network model concerns linked records rather than this column-oriented physical layout.

Primary note proof

Study-note section ADB41

> OLTP handles frequent operational transactions, such as account updates; OLAP scans and summarizes large datasets for analysis. A warehouse integrates data for historical analysis, often through extraction, transformation and loading. These are workload tendencies: OLTP can retain history and warehouses can be updated.

Study-note section ADB41

> Columnar storage groups values by column, reducing unnecessary column reads and aiding compression in scans; row-oriented storage groups each record's fields.

Study-note section ADB41

> Neither physical layout alone determines whether the logical model is relational.

Related reading (context, not primary note proof)

Study-note section ADB-S01

Question sources

Computer Science Exit Exam · Sene 2016 · Question 2-1

Computer Science Exit Exam practice collection · Question 69

Question 66 | SM0247

An online game wants extra read copies of player reference data near regional users while retaining the same logical data. Which concept specifically creates those additional copies, rather than dividing the only copy into fragments or choosing a model family?

A. Replication

B. NoSQL databases

C. Distributed databases

D. Partitioning

Hint

Separate copy placement, propagation timing and update-maintenance cost.

Answer A: A. Replication

Replication creates additional copies of selected data and can improve read locality by placing them near users. That directly satisfies the specified regional read-copy requirement. Distribution is the broader arrangement across sites, partitioning divides data into fragments, and NoSQL names a family of data models. Neither the broader architecture nor a model label specifically says that extra copies are created.

Why each choice is right or wrong

A - Correct: Replication creates the required regional read copies.

B - Incorrect: NoSQL selects a model family and does not itself create copies.

C - Incorrect: Distribution is broader than the requested copy-creation policy.

D - Incorrect: Partitioning divides data; by itself it creates no spare copy.

Primary note proof

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once. Copies improve read locality and can improve availability but require storage and update propagation. They can also spread mistakes, so replication does not replace independent backups.

Study-note section ADB36

> Partitioning improves locality and parallelism but alone creates no spare copy.

Study-note section ADB42

> Flexible schema does not mean no schema rules, and NoSQL does not imply no transactions.

Related reading (context, not primary note proof)

Study-note section ADB33

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 114 | lines 3-13

> 6.1 Distributed Database Concepts
> • A transaction can be executed by multiple networked computers in a unified manner.
> • A distributed database (DDB) processes Unit of execution (a transaction) in a distributed manner.
> • A distributed database (DDB) can be defined as :
> - A collection of multiple logically related database distributed over a computer network, and a distributed database management system as a software system that manages a distributed database while making the distribution transparent to the user.
> - The physical placement of data (files, relations, etc.) which is not known to the user (distribution transparency).

Fundamental of Database Systems module | PDF page 117 | lines 15-31

> 6.2 Data Replication and Fragmentation: Distributed data storage
> • There are two approaches to store the relation in the distributed database : Replication and Fragmentation
> I. Data Replication
> • The system maintain several identical copies of the relation & store each copy at a

different site
• In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead
II. Data Fragmentation
- Split a relation into logically related and correct parts.
- The main reasons for fragmenting a relation are
• Efficiency- data that is not needed by the local applications is not stored
• Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency
- but reconstruction of the whole relation will require accessing data from all sites containing part of the relation

Question sources

Computer Science Exit Exam · Sene 2016 · Question 15-3

Computer Science Exit Exam practice collection · Question 67

Question 67 | SM0298

In a generic DBMS component model applied to a MySQL deployment, which specialized role coordinates restoring valid data from stored backups and logs after data loss? Use generic functional roles, not claims about named MySQL engine internals.

A. Buffer Manager

B. Transaction Manager

C. Query Engine

D. Recovery Manager

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer D: D. Recovery Manager

Recovery is the specialized responsibility for restoring valid database state after failure or loss. Backups supply a separate stored copy, and available logs can restore later recoverable changes. The recovery-manager role therefore matches the question's generic component model. Buffer management caches working pages; transaction management is the broader parent responsibility, while the query engine processes requests. This makes no claim about MySQL's internal module names.

Why each choice is right or wrong

A - Incorrect: Buffers cache working data and are not durable backup copies.

B - Incorrect: Transaction management is broader than the specialized restoration role requested.

C - Incorrect: A query engine processes queries rather than coordinating restoration.

D - Correct: The generic recovery role coordinates restoration using durable recovery resources.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB26

> Generic recovery-manager terminology should not be mistaken for an exact MySQL component specification.

Study-note section ADB26

> A full backup copies the database to separate storage; scheduled backups plus archived logs support restoration to a chosen recoverable point.

Related reading (context, not primary note proof)

Study-note section ADB07

Study-note section ADB06

Study-note section ADB17

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 99 | lines 2-7

> ✓ Logging: In order to assure durability, every change in the database is logged separately on disk.
> ✓ Log manager initially writes the log in buffers and negotiates with the buffer manager to make sure that buffers are written to disk at appropriate times.
> ✓ Recovery Manager: will be able to examine the log of changes and restore the database to some consistent state.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 32-1

Computer Science Exit Exam practice collection · Question 70

Question 68 | SS102

Which listed sources can contain unstructured free text, images or audio? Select all that apply. Here Google search collections include free-text web pages; RDBMS systems may store text/BLOB attachments; the Apple Employee and Registrar student examples contain only fixed-field structured records.

A. Google search engine

B. RDBMS systems

C. Apple Employee database

D. Registrar student database

Hint

Classify the actual content or pattern being analyzed rather than the storage product.

Answer A, B: A. Google search engine; B. RDBMS systems

Unstructured classification depends on the actual content rather than the storage product. Search collections can contain free text and images, and an RDBMS can store text or binary attachments, so A and B qualify under the stated contents. The employee and registrar examples are explicitly restricted to fixed-field structured records, excluding the otherwise possible attachments that would change their classification.

Why each choice is right or wrong

A - Correct: The specified web collection includes unstructured free text.

B - Correct: The specified relational store may contain unstructured text/BLOB attachments.

C - Incorrect: This example is restricted to structured fixed-field records.

D - Incorrect: This example is restricted to structured fixed-field records.

Primary note proof

Study-note section ADB43

> Structured data follows explicit fields, such as a registrar's relational tables. Unstructured data includes free text, images and audio; web/search collections can include these alongside structured content. A DBMS can also store text or BLOBs, so a storage product does not prove all its contents are structured.

Study-note section ADB43

> Clustering groups similar cases without target labels; classification predicts categories and regression predicts numeric outcomes.

Study-note section ADB43

> If 75 of 100 shoe purchases include socks, confidence(shoes⇒socks)=0.75.

Question sources

Computer Science model exam collection · Question 9

Question 69 | SS123

Business people always remodel their business by closely studying their customers' purchasing habits over the course of their purchase history.
Accordingly, a salesperson at Markato shop noticed that people who always buy shoes will also buy socks with probability of 0.75 and hence decided
to put shoes and socks side by side for ease of accessibility. Which big data analytics technique could this salesperson could employ?

A. Association rule

B. Clustering algorithm

C. Predictive modeling

D. Regression modeling

Hint

Classify the actual content or pattern being analyzed rather than the storage product.

Answer A: Association-rule mining: shoes ⇒ socks has confidence 0.75.

The observation is a conditional co-purchase pattern: among shoe purchases, socks also appear with probability 0.75. ADB43 defines this as the confidence of an association rule and explicitly distinguishes co-occurrence mining from clustering, classification and numeric regression. The store-layout decision uses that rule; it does not change the underlying technique or prove causation.

Why each choice is right or wrong

A - Correct: The rule captures shoe/sock co-occurrence with conditional confidence 0.75.

B - Incorrect: Clustering groups similar cases without target labels rather than directly discovering this co-purchase rule.

C - Incorrect: The stated evidence is an association rule, not a separately specified predictive modeling task.

D - Incorrect: Regression predicts numeric outcomes rather than identifying this purchasing co-occurrence rule.

Primary note proof

Study-note section ADB43

> An association rule X⇒Y describes co-occurrence. Support=count(X and Y)/N; confidence=count(X and Y)/count(X)=P(Y|X). If 75 of 100 shoe purchases include socks, confidence(shoes⇒socks)=0.75. This does not prove causation or specify support without total transaction count. Clustering groups similar cases without target labels; classification predicts categories and regression predicts numeric outcomes. Purchasing co-occurrence is association mining even when its result informs store layout.

Question sources

Computer Science model exam collection · Question 30

Question 70 | SS130

Using this convention, Model Testing means measuring a trained model on held-out test data; Model Evaluation means the subsequent release decision using those results and business success criteria. Which sequence applies before deploying this newly trained model?

A. Model Training, Model Evaluation, Model Testing, Model Deployment

B. Model Training, Model Testing, Model Deployment, Model Evaluation

C. Model Training, Model Testing, Model Evaluation, Model Deployment

D. Model Evaluation, Model Training , Model Testing, Model Deployment

Hint

Identify the objective and the stated meanings of the workflow stages.

Answer C: C. Model Training, Model Testing, Model Evaluation, Model Deployment

Under the stated convention, training first produces the model. Testing then measures it on held-out data, and evaluation uses those results to decide whether business criteria justify deployment. C follows that sequence. Evaluation during model selection can use a different ordering, but this question explicitly assigns evaluation to the subsequent release review, so the other sequences misplace that review.

Why each choice is right or wrong

A - Incorrect: It puts the specified post-test release review before testing.

B - Incorrect: It deploys before the required release review.

C - Correct: Training, held-out testing, release evaluation and deployment follow the defined roles.

D - Incorrect: It evaluates the newly trained model before that model has been trained.

Primary note proof

Study-note section ADB44

> CRISP-DM starts with business understanding: define the decision and success criteria. Then understand data, prepare it, model, evaluate and deploy; the process is iterative. Cleaning, integration and feature selection occur during preparation, not before the business problem exists.

Study-note section ADB44

> If an MCQ asks training→testing→evaluation→deployment, state that convention; another valid workflow can evaluate during model selection before final testing.

Study-note section ADB44

> Within modeling, split data appropriately, train on training data, tune using validation and evaluate once on a held-out test set before deployment.

Question sources

Computer Science model exam collection · Question 37

Question 71 | SS188

Identify the one that comes first in data mining process

A. Business understanding

B. Data integration

C. Data cleaning

D. Data selection

Hint

Identify the objective and the stated meanings of the workflow stages.

Answer A: Business understanding.

ADB44 starts the data-mining workflow by defining the business decision and its success criteria. This establishes why data will be collected and modeled. Integration, cleaning and feature selection prepare data after the problem has been understood; they do not replace the initial business-understanding stage. Thus A precedes the other listed activities.

Why each choice is right or wrong

A - Correct: Business understanding first defines the decision to support and the criteria for success.

B - Incorrect: Data integration prepares sources after the business problem has been identified.

C - Incorrect: Cleaning improves data quality during preparation rather than defining the initial business objective.

D - Incorrect: Selecting relevant data depends on an already understood business problem and modeling objective.

Primary note proof

Study-note section ADB44

> CRISP-DM starts with business understanding: define the decision and success criteria. Then understand data, prepare it, model, evaluate and deploy; the process is iterative. Cleaning, integration and feature selection occur during preparation, not before the business problem exists.

Study-note section ADB44

> Deployment includes monitoring and may trigger another preparation/modeling cycle.

Study-note section ADB44

If an MCQ asks training→testing→evaluation→deployment, state that convention; another valid workflow can evaluate during model selection before final testing.

Question sources

Computer Science model exam collection · Question 95

Question 72 | AM0162

What is the defining purpose of a transaction in database management?

A. To secure the database from unauthorized access

B. To optimize query performance

C. To group database operations into one logical unit of work

D. To define the structure and organization of a database

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer C: C. To group database operations into one logical unit of work

A transaction groups database operations into one logical unit of work. The DBMS supports its atomicity, consistency, isolation and durability requirements, including controlled interference with concurrent transactions. Grouping is the defining concept, rather than query tuning, access authorization or schema construction. Read-only work can also form a transaction, so the definition is not limited to operations that change stored data.

Why each choice is right or wrong

A - Incorrect: Authorization governs who may access data rather than defining transaction grouping.

B - Incorrect: Optimization chooses query execution strategies.

C - Correct: A transaction is one logical unit of database work.

D - Incorrect: Schema definition describes data structures rather than a unit of work.

Primary note proof

Study-note section ADB06

A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Study-note section ADB11

A transaction must own the appropriate lock before access and can release only its own locks.

Study-note section ADB11

One transaction may request a lock on an already locked item: it is granted if compatible, otherwise waits or aborts under the policy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 63 | lines 3-9

3.1 Introduction to Transaction
- A Transaction:
  - Logical unit of database processing that includes one or more access operations (read -retrieval, write - insert or update, delete).
- Examples include ATM transactions, credit card approvals, flight reservations, hotel check-in, phone calls, supermarket scanning, academic registration and billing.
- Transaction boundaries:

Fundamental of Database Systems module | PDF page 66 | lines 4-13

- Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
- Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.
- Isolation: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary
- Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.

Example

Question sources

Computer Science MCQ practice collection · Question 162

Question 73 | AM0177

A correctly implemented financial transaction must preserve a declared total-balance invariant when it completes. Which ACID property names preservation of such integrity rules? Focus on the invariant rather than partial execution, concurrent interference or survival after a crash.

A. Atomicity

B. Consistency

C. Isolation

D. Durability

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: B. Consistency

Consistency means that correct transactional work preserves declared integrity rules. Maintaining the total-balance invariant is therefore the consistency requirement. Atomicity addresses whether the effects happen together, isolation controls interference among concurrent transactions and durability preserves committed effects after failure. These properties cooperate, but the revised question identifies invariant preservation as the particular guarantee to name.

Why each choice is right or wrong

A - Incorrect: Atomicity names all-or-none effects rather than invariant preservation itself.

B - Correct: Consistency names preservation of declared integrity rules by correct work.

C - Incorrect: Isolation controls concurrent interference.

D - Incorrect: Durability preserves committed effects after failure.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 66 | lines 4-13

> • Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
> • Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.
> • Isolation: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary
> • Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.
> Example

Question sources

Computer Science MCQ practice collection · Question 177

Question 74 | AM0185

In the object-oriented model, an object is defined as:

A. An instance of a class

B. A collection of attributes

C. A primary key in a database

D. A relationship between tables

Hint

Distinguish an object instance from its class, attribute values and persistent identity.

Answer A: An instance of a class.

ADB01 explicitly distinguishes a class specification from each object instantiated from it. A class defines available attributes and operations, while an object is one individual instance with identity, state and behavior. A collection of attributes omits identity and behavior; a key or an intertable relationship belongs to a different relational description.

Why each choice is right or wrong

A - Correct: An object is an individual instance of a class with its own identity and state.

B - Incorrect: Attributes describe state but omit the identity and behavior that also characterize an object.

C - Incorrect: A database primary key is an identifier constraint, not the definition of an object instance.

D - Incorrect: A relationship between tables is a relational association rather than an object instance.

Primary note proof

Study-note section ADB01

> An object combines identity, state and behavior. A class specifies the attributes and operations of its instances; an object is one instance. The object identifier (OID) distinguishes objects independently of their attribute values and remains stable over that object's lifetime. Two objects can have equal values and different identities; changing a name does not create a new identity.

Study-note section ADB01

> Complex CAD or multimedia structures can suit an object model, but many-to-many relationships alone do not require abandoning relational design.

Study-note section ADB01

> An object-relational DBMS extends a relational system with richer types; ORM maps programming objects into relational tables.

Question sources

Computer Science MCQ practice collection · Question 185

Question 75 | AM0186

In the object-oriented model, encapsulation refers to:

A. Combining multiple classes into a single entity

B. Hiding the internal details of an object

C. Establishing relationships between objects

D. Defining the inheritance hierarchy

Hint

Separate the object boundary, the exposed contract and links between objects.

Answer B: Hiding and controlling access to internal object details.

ADB02 describes encapsulation as grouping state with operations and controlling access to implementation details. A withdrawal method can enforce rules while hiding storage. B states the relevant information-hiding aspect. Combining classes, associating objects and defining subtype hierarchies are distinct activities; none describes the encapsulation boundary around an object's state and operations.

Why each choice is right or wrong

A - Incorrect: Encapsulation groups an object's state and operations, not multiple classes into one entity.

B - Correct: Encapsulation controls access to the internal representation behind the object's operations.

C - Incorrect: Links between objects are associations rather than encapsulation itself.

D - Incorrect: Superclass/subclass structure concerns inheritance rather than information-hiding boundaries.

Primary note proof

Study-note section ADB02

> Encapsulation groups state with operations and controls access to implementation details. Account.withdraw(amount) can enforce a balance rule while hiding its storage. Abstraction presents the relevant contract-what withdrawal does-without exposing how it is implemented. These ideas overlap in information hiding, but combining state and methods specifically describes encapsulation.

Study-note section ADB02

> A class defines structure and behavior; an attribute describes state; a method implements an operation.

Study-note section ADB02

It is different from inheritance, which means an is-a relationship between types.

Question sources

Computer Science MCQ practice collection · Question 186

Question 76 | AM0187

In the object-oriented model, inheritance allows:

A. Objects to have multiple parents

B. Objects to inherit attributes and behaviors from a superclass

C. Objects to be created dynamically at runtime

D. Objects to establish relationships with other objects

Hint

Distinguish the subtype relationship from use through a common interface.

Answer B: Inherit applicable attributes and behavior from a superclass.

ADB03 defines inheritance as deriving a subtype from a superclass and reusing its applicable attributes and behavior. B captures that idea. It does not inherently require multiple parents, dynamically create objects, or merely link unrelated objects. Those alternatives concern particular inheritance forms, instantiation or association rather than the general meaning of inheritance.

Why each choice is right or wrong

A - Incorrect: Multiple parents are not a general requirement or guarantee of inheritance.

B - Correct: A subtype inherits applicable superclass attributes and behavior.

C - Incorrect: Dynamic object creation is instantiation, not the capability defined by inheritance.

D - Incorrect: Relationships between separate objects are associations rather than the is-a subtype relationship.

Primary note proof

Study-note section ADB03

Inheritance derives a subtype from a superclass, reusing applicable attributes and behavior. GraduateStudent is a Student; association instead says Student belongs to Department. EER teaches inherited attributes and relationships, while a programming object model also specifies behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 13 | lines 5-9

Subclasses and super classes can be thought as useful ways to describe similarities in the entities represented. Because subclasses inherit their attributes and relationships from the superclass, the
database designer can represent complex, hierarchical interdependencies (inheritance) within the
system being described.
Specialization is the process of dividing a parent-level entity (super-class) into narrower

Question sources

Computer Science MCQ practice collection · Question 187

Question 77 | AM0188

Which concept in the object-oriented model allows different classes to share the same method name but with different implementations?

A. Polymorphism

B. Abstraction

C. Encapsulation

D. Inheritance

Hint

Distinguish the subtype relationship from use through a common interface.

Answer A: Polymorphism.

ADB03 illustrates Shape.area dispatching to Circle.area or Rectangle.area according to the actual object. The common operation name has different compatible implementations, which is polymorphism. Inheritance can provide the subtype structure used by that mechanism, but is not itself the runtime dispatch behavior. Abstraction and encapsulation concern contracts and implementation access instead.

Why each choice is right or wrong

A - Correct: Polymorphism lets the same interface operation invoke different implementations for compatible object types.

B - Incorrect: Abstraction exposes a relevant contract rather than selecting among differing method implementations.

C - Incorrect: Encapsulation controls internal access rather than providing type-dependent method dispatch.

D - Incorrect: Inheritance supplies a structural subtype relation; polymorphism describes the differing implementations through a common interface.

Primary note proof

Study-note section ADB03

> Subtype polymorphism lets a superclass/interface reference denote objects of different compatible subclasses. The same operation can dispatch to different implementations: Shape.area() calls Circle.area() or Rectangle.area() according to the actual object. This is not arbitrary conversion of an object into an unrelated class, nor dynamic class mutation. Inheritance establishes a subtype relationship; polymorphism uses that relationship through a common interface. Distinguish inheritance as a structural relationship from polymorphism as a use of that relationship at runtime.

Question sources

Computer Science MCQ practice collection · Question 188

Question 78 | AM0189

The process of combining data and methods into a single entity is known as:

A. Inheritance

B. Polymorphism

C. Abstraction

D. Encapsulation

Hint

Separate the object boundary, the exposed contract and links between objects.

Answer D: Encapsulation.

The question specifically asks about placing data and methods inside one entity. ADB02 identifies that grouping as encapsulation, along with controlled access to implementation details. Abstraction concerns what the contract exposes; inheritance derives compatible subtypes; polymorphism invokes different implementations through a common interface. None of those replaces the state-and-method grouping named in the stem.

Why each choice is right or wrong

A - Incorrect: Inheritance derives a subtype and reuses superclass features rather than naming state/method grouping.

B - Incorrect: Polymorphism varies implementations behind a compatible common interface rather than grouping data and methods.

C - Incorrect: Abstraction presents a relevant contract; the specific grouping asked for is encapsulation.

D - Correct: Encapsulation packages state and operations together within the object boundary.

Primary note proof

Study-note section ADB02

> Encapsulation groups state with operations and controls access to implementation details. Account.withdraw(amount) can enforce a balance rule while hiding its storage. Abstraction presents the relevant contract-what withdrawal does-without exposing how it is implemented. These ideas overlap in information hiding, but combining state and methods specifically describes encapsulation.

Study-note section ADB02

> It is different from inheritance, which means an is-a relationship between types.

Study-note section ADB02

> A class defines structure and behavior; an attribute describes state; a method implements an operation.

Question sources

Computer Science MCQ practice collection · Question 189

Question 79 | AM0190

The object-oriented model supports the concept of association, which represents:

A. Inheritance relationships between classes

B. Shared attributes among objects

C. The interactions between objects

D. Data duplication in the database

Hint

Separate the object boundary, the exposed contract and links between objects.

Answer C: Links and interactions between objects.

ADB02 defines association as links between objects, commonly implemented with references, and describes methods invoking operations on other objects. C is the closest description of that collaboration. An association is not an is-a inheritance hierarchy, shared attribute definitions or duplication. For example, Student belonging to Department links objects without making either a subtype of the other.

Why each choice is right or wrong

A - Incorrect: Inheritance is an is-a relationship between types, distinct from association links between objects.

B - Incorrect: Having similar attributes does not itself define an association between particular objects.

C - Correct: Associations link objects and support their interactions or collaboration through references.

D - Incorrect: Association represents relationships rather than duplicated copies of the same database facts.

Primary note proof

Study-note section ADB02

> An association links objects, often through references; Department.employees can reference Employee objects. It is different from inheritance, which means an is-a relationship between types. Methods can invoke operations on other objects, supporting collaboration. A class defines structure and behavior; an attribute describes state; a method implements an operation. Do not treat each of these as an interchangeable name for the whole object.

Question sources

Computer Science MCQ practice collection · Question 190

Question 80 | AM0191

Circle and Rectangle already inherit from Shape. A Shape reference can hold either object and dispatch area() to its actual implementation at runtime. Which concept names this common-interface use, rather than the structural subtype relationship?

A. Polymorphism

B. Inheritance

C. Abstraction

D. Encapsulation

Hint

Distinguish the subtype relationship from use through a common interface.

Answer A: A. Polymorphism

Subtype polymorphism allows a common superclass or interface reference to denote different compatible subclass objects. Dispatching area() according to whether the actual object is a Circle or Rectangle is that runtime use. Inheritance establishes the subtype relationship already given in the stem. Abstraction exposes a contract and encapsulation controls representation, but neither names this particular substitutable runtime dispatch.

Why each choice is right or wrong

A - Correct: Using one superclass interface with subclass-specific dispatch is subtype polymorphism.

B - Incorrect: Inheritance is the established structural prerequisite, not the runtime use requested.

C - Incorrect: Abstraction describes a contract rather than the dispatch behavior.

D - Incorrect: Encapsulation controls internal representation and access.

Primary note proof

Study-note section ADB03

> Subtype polymorphism lets a superclass/interface reference denote objects of different compatible subclasses. The same operation can dispatch to different implementations: Shape.area() calls Circle.area() or Rectangle.area() according to the actual object. This is not arbitrary conversion of an object into an unrelated class, nor dynamic class mutation. Inheritance establishes a subtype relationship; polymorphism uses that relationship through a common interface. Distinguish inheritance as a structural relationship from polymorphism as a use of that relationship at runtime.

Study-note section ADB02

> Encapsulation groups state with operations and controls access to implementation details.

Study-note section ADB02

These ideas overlap in information hiding, but combining state and methods specifically describes encapsulation.

Question sources

Computer Science MCQ practice collection · Question 191

Question 81 | AM0192

What is the purpose of abstraction in the object-oriented model?

A. To combine multiple classes into a single entity

B. To hide the internal details of an object

C. To establish relationships between objects

D. To define the attributes and behaviors of a class

Hint

Separate the object boundary, the exposed contract and links between objects.

Answer B: Hide implementation details while exposing a relevant contract.

ADB02 explains abstraction through an operation's relevant contract: what withdrawal does is exposed without showing how storage implements it. B captures hiding those irrelevant internal details. This overlaps encapsulation, but abstraction emphasizes the view or contract. Combining classes, creating associations or listing class members is not its defining purpose.

Why each choice is right or wrong

A - Incorrect: Abstraction does not mean merging several classes into one entity.

B - Correct: Abstraction exposes relevant behavior while hiding implementation details from the client view.

C - Incorrect: Establishing links between objects is association rather than abstraction's defining purpose.

D - Incorrect: A class specifies attributes and behavior; abstraction chooses the relevant contract exposed to users.

Primary note proof

Study-note section ADB02

Encapsulation groups state with operations and controls access to implementation details. Account.withdraw(amount) can enforce a balance rule while hiding its storage. Abstraction presents the relevant contract-what withdrawal does-without exposing how it is implemented. These ideas overlap in information hiding, but combining state and methods specifically describes encapsulation.

Study-note section ADB03

EER teaches inherited attributes and relationships, while a programming object model also specifies behavior.

Study-note section ADB02

An association links objects, often through references; Department.employees can reference Employee objects.

Question sources

Computer Science MCQ practice collection · Question 192

Question 82 | AM0193

The concept of polymorphism allows objects to:

A. Inherit attributes from multiple classes

B. Change their class dynamically at runtime

C. Access private methods of other objects

D. Be treated as objects of different classes

Hint

Distinguish the subtype relationship from use through a common interface.

Answer D: Use an object through a compatible superclass or interface type.

ADB03 allows a superclass or interface reference to denote compatible subclass objects. D is correct in that sense: the same object can be treated through its actual type and a compatible supertype. This does not permit arbitrary conversion to unrelated classes. Multiple inheritance, runtime class mutation and access to another object's private methods are different concepts and are not granted by polymorphism.

Why each choice is right or wrong

A - Incorrect: Inheriting from multiple classes is multiple inheritance, not the operation described by polymorphism.

B - Incorrect: Polymorphism does not change the actual class of an object dynamically.

C - Incorrect: Polymorphism does not bypass private-access restrictions on other objects.

D - Correct: An object can be used through compatible superclass/interface types; this does not mean arbitrary unrelated classes.

Primary note proof

Study-note section ADB03

> Subtype polymorphism lets a superclass/interface reference denote objects of different compatible subclasses. The same operation can dispatch to different implementations: Shape.area() calls Circle.area() or Rectangle.area() according to the actual object. This is not arbitrary conversion of an object into an unrelated class, nor dynamic class mutation. Inheritance establishes a subtype relationship; polymorphism uses that relationship through a common interface. Distinguish inheritance as a structural relationship from polymorphism as a use of that relationship at runtime.

Question sources

Computer Science MCQ practice collection · Question 193

Question 83 | AM0194

Which concept in the object-oriented model defines the structure and behavior of objects?

A. Class

B. Object

C. Attribute

D. Method

Hint

Distinguish an object instance from its class, attribute values and persistent identity.

Answer A: Class.

A class is the specification that defines attributes and operations for its instances, as ADB01 states. An object is an instance conforming to that specification. An attribute accounts for part of state and a method for one operation, but neither individually defines the whole structure and behavior of the object type.

Why each choice is right or wrong

A - Correct: A class specifies the structure and available operations for its object instances.

B - Incorrect: An object is one instance of the specification rather than the specification itself.

C - Incorrect: An attribute represents part of an object's state rather than its whole structure and behavior.

D - Incorrect: A method implements an operation but does not alone define the complete object type.

Primary note proof

Study-note section ADB01

> An object combines identity, state and behavior. A class specifies the attributes and operations of its instances; an object is one instance. The object identifier (OID) distinguishes objects independently of their attribute values and remains stable over that object's lifetime. Two objects can have equal values and different identities; changing a name does not create a new identity.

Study-note section ADB01

> Complex CAD or multimedia structures can suit an object model, but many-to-many relationships alone do not require abandoning relational design.

Study-note section ADB01

> An object-relational DBMS extends a relational system with richer types; ORM maps programming objects into relational tables.

Question sources

Computer Science MCQ practice collection · Question 194

Question 84 | AM0195

Which recovery method aims to restore the database to a previous consistent state by undoing the effects of incomplete or failed transactions?

A. Rollback recovery

B. Redo recovery

C. Checkpoint recovery

D. Restart recovery

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer A: Rollback recovery

Undo restores before images and removes after images; failed transactions that updated the database must roll back (p.107). This directly answers the specific action, although undo may be one phase of broader crash recovery.

Why each choice is right or wrong

A - Correct: Rollback is the direct undo operation.

B - Incorrect: Redo restores after images instead.

C - Incorrect: Checkpoints synchronize database and log; they do not name the undo action.

D - Incorrect: A restart can include undo, but is broader than the requested failed-transaction reversal.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Study-note section ADB30

> Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 195

Question 85 | AM0196

Which recovery method involves reapplying the changes made by committed transactions from a log file to restore the database?

A. Redo recovery

B. Checkpoint recovery

C. Restart recovery

D. Backup recovery

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer A: Redo recovery

If a transaction committed but its buffers did not reach disk, recovery redoes its updates for durability (p.103). Redo restores after images (p.107), whose log records are required before commit (p.104).

Why each choice is right or wrong

A - Correct: Redo is exactly reapplying logged committed updates.

B - Incorrect: A checkpoint reduces the amount of recovery work; it is not the replay operation.

C - Incorrect: Restart recovery can invoke redo but names a larger process.

D - Incorrect: A backup restores a stored database copy, not this specific logged replay.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB30

> Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Study-note section ADB30

> A checkpoint is neither a full backup nor an automatic commit of active transactions.

Related reading (context, not primary note proof)

Study-note section ADB26

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 104 | lines 19-25

• For Undo: Before a data item‘s AFIM is flushed to the database disk (overwriting the BFIM) its BFIM must be written to the log and the log must be saved on a stable store (log disk).
• For Redo: Before a transaction executes its commit operation, all its AFIMs must be written to the log and the log must be saved on a stable store.
• Standard Recovery Terminology

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 196

Question 86 | AM0197

Which recovery method involves creating checkpoints at regular intervals to save the current state of the database and transaction logs?

A. Checkpoint recovery

B. Redo recovery

C. Rollback recovery

D. Restart recovery

Hint

Identify which history a recovery boundary can safely exclude and whether active work spans it.

Answer A: Checkpoint recovery

Checkpointing synchronizes database and log (p.105); its listed steps force modified buffers to disk, write a checkpoint record, and save the log (p.107). This is not necessarily a full database backup.

Why each choice is right or wrong

A - Correct: The named synchronization procedure matches.

B - Incorrect: Redo reapplies after images, not periodic synchronization.

C - Incorrect: Rollback restores before images.

D - Incorrect: Restart follows failure, whereas checkpoints are established during operation.

Primary note proof

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history. In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo. Active transactions can span the checkpoint, so their relevant earlier updates can still need undo. Do not assume every transaction before a checkpoint needs no processing.

Study-note section ADB27

> To acknowledge a durable commit, force the transaction's required log records, including its commit record, to stable storage.

Study-note section ADB27

> Write-ahead logging requires the update log needed for recovery to reach stable storage before the corresponding changed data page reaches disk.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 105 | lines 25-36

Check pointing
• Log file is used to recover failed DB but we may not know how far back in the log to search. Thus
• Checkpoint is a Point of synchronization between database and log file.
• Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.
• When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 197

Question 87 | AM0198

Which recovery method involves restarting the system and recovering the database by applying the necessary redo and undo operations?

A. Restart recovery

B. Checkpoint recovery

C. Redo recovery

D. Rollback recovery

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer A: Restart recovery

Recovery from a non-catastrophic crash may combine undo and redo (pp.103,107). A is the best descriptive name for restarting and performing that combined recovery, although the module does not define four mutually exclusive method classes under these exact option names.

Why each choice is right or wrong

A - Correct: Restart recovery describes the complete post-crash recovery process stated in the stem.

B - Incorrect: Checkpointing supplies a saved starting boundary rather than the whole restart process.

C - Incorrect: Redo supplies only the forward part of the stated combination.

D - Incorrect: Rollback supplies only the undo part.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB26

> These are cooperating mechanisms, not four mutually exclusive complete alternatives.

Study-note section ADB27

> Redo installs logged after-images when necessary, usually forward.

Related reading (context, not primary note proof)

Study-note section ADB30

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Fundamental of Database Systems module | PDF page 110 | lines 5-13

◦ Repeating history during redo:
• ARIES will retrace all actions of the database system prior to the crash to reconstruct the database state when the crash occurred.
• Transaction that were un committed at the time of crash are undone
◦ Logging changes during undo:

• It will prevent ARIES from repeating the completed undo operations if a failure occurs during recovery, which causes a restart of the recovery process.
Data Update : Four types

Question sources

Computer Science MCQ practice collection · Question 198

Question 88 | AM0199

Which recovery method involves restoring the database from a backup copy taken prior to the failure?

A. Backup recovery

B. Rollback recovery

C. Redo recovery

D. Restart recovery

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer A: Backup recovery

For catastrophic damage the module restores a past database copy from backup storage and reconstructs committed operations from the backup log (p.103). A names restoration of that backup copy; a backup alone may need subsequent log replay.

Why each choice is right or wrong

A - Correct: Restoration from a stored copy is the stated backup operation.

B - Incorrect: Rollback reverses updates using before images, not a backup copy.

C - Incorrect: Redo replays logged changes, which may supplement backup restoration.

D - Incorrect: Restart names an overall recovery process rather than its backup-copy source.

Primary note proof

Study-note section ADB26

Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB26

Rollback handles failed/uncommitted work; redo preserves committed work missing from pages; checkpoints reduce log-search work; backups provide another stored copy.

Study-note section ADB27

Redo installs logged after-images when necessary, usually forward.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Question sources

Computer Science MCQ practice collection · Question 199

Question 89 | AM0200

Which recovery method is used to recover the database after a media failure, such as a disk crash?

A. Backup recovery

B. Redo recovery

C. Checkpoint recovery

D. Rollback recovery

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer A: Backup recovery

Disk-head crash is media failure (p.97). Where that destroys database contents, the catastrophic-recovery strategy restores a past backup and then reconstructs committed operations from the backup log (p.103).

Why each choice is right or wrong

A - Correct: A backup supplies contents lost in the media failure.

B - Incorrect: Redo can advance a restored copy but alone assumes data pages available to update.

C - Incorrect: A checkpoint is not a replacement copy of a destroyed database.

D - Incorrect: Undo cannot recreate arbitrary destroyed pages by reversing incomplete transactions alone.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB30

> Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Related reading (context, not primary note proof)

Study-note section ADB27

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 97 | lines 4-13

> i. Purpose of Database Recovery
> ◦ To bring the database into the last consistent state, which existed prior to the failure.
> ◦ To preserve transaction properties (Atomicity &and Durability).
> • The recovery manager of a DBMS is responsible to ensure
> • atomicity by undoing the action of transaction that do not commit
> • Durability by making sure that all actions of committed transaction survive system crash

ii. Types of Failure
◦ The database may become unavailable for use due to :

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Question sources

Computer Science MCQ practice collection · Question 200

Question 90 | AM0201

A crash-recovery system has a long log history and wants to reduce the log-search work compared with scanning all history. Which method adds durable recovery-state records for that purpose?

A. Redo recovery

B. Checkpoint recovery

C. Backup recovery

D. Rollback recovery

Hint

Identify which history a recovery boundary can safely exclude and whether active work spans it.

Answer B: B. Checkpoint recovery

Checkpoints record durable recovery state so recovery need not inspect all old history. That is the requested mechanism for reducing log-search work. Redo reinstalls missing changes, rollback removes incomplete effects and backup recovery starts from a separate copy. They may cooperate with checkpoints, but their defining actions do not name the added recovery-state boundary specified in this question.

Why each choice is right or wrong

A - Incorrect: Redo applies logged after-images rather than defining the recovery-state boundary.

B - Correct: Checkpoint information reduces how much old history must be searched.

C - Incorrect: A backup is a separate stored copy rather than the stated history-bounding record.

D - Incorrect: Rollback reverses incomplete effects rather than adding this recovery-state boundary.

Primary note proof

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history. In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo. Active transactions can span the checkpoint, so their relevant earlier updates can still need undo. Do not assume every transaction before a checkpoint needs no processing.

Study-note section ADB26

> Rollback handles failed/uncommitted work; redo preserves committed work missing from pages; checkpoints reduce log-search work; backups provide another stored copy.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Related reading (context, not primary note proof)

Study-note section ADB27

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

> • Recovery from catastrophic
> • If there is extensive damage to the wide portion of the database
> • This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
> • Recovery from non-catastrophic failure
> • When the database is not physically damaged but has be come in

consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 105 | lines 25-36

Check pointing
• Log file is used to recover failed DB but we may not know how far back in the log to search. Thus
• Checkpoint is a Point of synchronization between database and log file.
• Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.
• When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 201

Computer Science MCQ practice collection · Question 211

Question 91 | AM0202

Which recovery method is focused on ensuring that all committed changes are durable and can survive system failures?

A. Redo recovery

B. Restart recovery

C. Rollback recovery

D. Checkpoint recovery

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer A: Redo recovery

Durability requires committed effects to survive crashes (p.97). The module specifically redoes committed updates lost between commit and buffer flushing (p.103), making redo the direct mechanism asked for.

Why each choice is right or wrong

A - Correct: Redo reapplies the missing committed updates.

B - Incorrect: Restart is a broader process that can invoke redo and undo.

C - Incorrect: Rollback preserves atomicity by removing uncommitted effects.

D - Incorrect: Checkpoints reduce replay work but are not the committed-update replay operation.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB26

> Rollback handles failed/uncommitted work; redo preserves committed work missing from pages; checkpoints reduce log-search work; backups provide another stored copy.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Related reading (context, not primary note proof)

Study-note section ADB30

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 97 | lines 4-13

i. Purpose of Database Recovery
◦ To bring the database into the last consistent state, which existed prior to the failure.
◦ To preserve transaction properties (Atomicity &and Durability).
• The recovery manager of a DBMS is responsible to ensure
• atomicity by undoing the action of transaction that do not commit
• Durability by making sure that all actions of committed transaction survive system crash
ii. Types of Failure
◦ The database may become unavailable for use due to :

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Question sources

Computer Science MCQ practice collection · Question 202

Question 92 | AM0203

Which recovery method uses the transaction log to identify the changes made by transactions and undo the effects of incomplete or failed transactions?

A. Rollback recovery

B. Redo recovery

C. Checkpoint recovery

D. Restart recovery

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer A: Rollback recovery

Before images logged under WAL permit undo of failed transaction effects (p.104); rollback restores them (p.107). The stem specifies this particular operation rather than all crash-recovery phases.

Why each choice is right or wrong

A - Correct: Rollback is the direct log-based undo operation.

B - Incorrect: Redo restores new values instead of removing failed updates.

C - Incorrect: Checkpointing records a synchronization boundary.

D - Incorrect: Restart can include rollback, but is broader than the named reversal.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Study-note section ADB30

> Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 104 | lines 19-25

• For Undo: Before a data item‘s AFIM is flushed to the database disk (overwriting the BFIM) its BFIM must be written to the log and the log must be saved on a stable store (log disk).
• For Redo: Before a transaction executes its commit operation, all its AFIMs must be written to the log and the log must be saved on a stable store.
• Standard Recovery Terminology

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 203

Question 93 | AM0204

Which recovery method is used to recover the database after a system crash or power failure?

A. Restart recovery

B. Backup recovery

C. Redo recovery

D. Checkpoint recovery

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer A: Restart recovery

For a recoverable system crash, recovery uses undo/redo to restore consistency (pp.103,107). A is the broad descriptive answer for restarting the failed system. If power failure also destroys media, backup restoration may additionally be required; the question supplies no such destruction.

Why each choice is right or wrong

A - Correct: Restart recovery denotes the overall post-crash process.

B - Incorrect: Backup is required for damaged/lost storage, which is not specified here.

C - Incorrect: Redo alone may be insufficient when incomplete transactions reached disk.

D - Incorrect: A checkpoint assists recovery but does not itself execute the full crash recovery.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB30

> Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Related reading (context, not primary note proof)

Study-note section ADB27

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 98 | lines 4-31

- What causes a Transaction to fail

1. A computer failure (system crash):
A hardware or software error occurs in the computer system during transaction execution. If the hardware crashes, the contents of the computer‘s internal memory may be lost.
2. A transaction or system error:
Some operation in the transaction may cause it to fail, such as integer overflow or division by zero. Transaction failure may also occur because of erroneous parameter values or because of a logical programming error. In addition, the user may interrupt the transaction during its execution.
3. Exception conditions detected by the transaction:
◦ Certain conditions forces cancellation of the transaction.
◦ For example, data for the transaction may not be found. such as insufficient account balance in a banking database, may cause a transaction, such as a fund withdrawal from that account, to be canceled.
4. Concurrency control enforcement:
The concurrency control method may decide to abort the transaction, to be restarted later, because it violates serializability or because several transactions are in a state of deadlock (see Chapter 2).
5. Disk failure:
Some disk blocks may lose their data because of a read or write malfunction or because of a disk read/write head crash. This may happen during a read or a write operation of the transaction.
6. Physical problems and catastrophes:
This refers to an endless list of problems that includes power or air-conditioning failure, fire, theft, overwriting disks or tapes by mistake
Transaction Manager : Accepts transaction commands from an application, which tell the transaction manager when transactions begin and end, as well as information about the expectations of the application.
The transaction processor performs the following tasks:

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore

to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 204

Question 94 | AM0205

What is the primary goal of database recovery after a failure?

A. Restoring the database to a previous state

B. Preventing future failures from occurring

C. Optimizing database performance

D. Enforcing data integrity constraints

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer A: Restoring the database to a previous state

The explicit purpose is bringing the database into its last consistent state before failure (p.97). A is correct only with this consistency qualification; restoring an arbitrary earlier corrupt state is not the goal.

Why each choice is right or wrong

A - Correct: Read previous state as the last consistent state, as stated.

B - Incorrect: Recovery handles failures rather than preventing all future ones.

C - Incorrect: Performance is not the stated recovery purpose.

D - Incorrect: Integrity is relevant, but constraint enforcement is not the direct restoration objective stated here.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Study-note section ADB26

> Generic recovery-manager terminology should not be mistaken for an exact MySQL component specification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 97 | lines 4-13

> i. Purpose of Database Recovery
> ◦ To bring the database into the last consistent state, which existed prior to the failure.
> ◦ To preserve transaction properties (Atomicity &and Durability).
> • The recovery manager of a DBMS is responsible to ensure
> • atomicity by undoing the action of transaction that do not commit
> • Durability by making sure that all actions of committed transaction survive system crash
> ii. Types of Failure
> ◦ The database may become unavailable for use due to :

Question sources

Computer Science MCQ practice collection · Question 205

Question 95 | AM0206

Which recovery method involves periodically making a copy of the entire database onto a separate storage medium?

A. Rollback recovery

B. Checkpoint recovery

C. Restart recovery

D. Backup recovery

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer D: Backup recovery

A full backup copies the database to separate storage so it can later be restored, often followed by replay of archived logs. Rollback reverses failed transaction work; restart recovery restores a valid state after a crash. A checkpoint records recovery state to reduce log-search and recovery work and is not a full independent database copy. The question's periodic separate copy is therefore the backup mechanism, represented by backup recovery.

Why each choice is right or wrong

A - Incorrect: Incorrect: rollback reverses transaction effects rather than making an independent database copy.

B - Incorrect: Incorrect: checkpointing records recovery state and limits recovery work; it is not a full backup.

C - Incorrect: Incorrect: restart recovery is the crash-recovery process, not the periodic creation of a separate copy.

D - Correct: Correct: backup recovery relies on a stored independent database copy.

Primary note proof

Study-note section ADB26

> A full backup copies the database to separate storage; scheduled backups plus archived logs support restoration to a chosen recoverable point.

Study-note section ADB26

> Rollback handles failed/uncommitted work; redo preserves committed work missing from pages; checkpoints reduce log-search work; backups provide another stored copy.

Study-note section ADB30

> A checkpoint is neither a full backup nor an automatic commit of active transactions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 105 | lines 25-36

Check pointing
• Log file is used to recover failed DB but we may not know how far back in the log to search. Thus
• Checkpoint is a Point of synchronization between database and log file.
• Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.
• When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Question sources

Computer Science MCQ practice collection · Question 206

Question 96 | AM0207

Which recovery method involves undoing the effects of incomplete or failed transactions?

A. Rollback recovery

B. Checkpoint recovery

C. Restart recovery

D. Backup recovery

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer A: Rollback recovery

The module calls undoing a failed transaction rollback and specifies restoring before images (p.107). A therefore directly names removal of incomplete effects.

Why each choice is right or wrong

A - Correct: Rollback is the named undo operation.

B - Incorrect: Checkpoint is synchronization, not transaction reversal.

C - Incorrect: Restart can include undo but is the wider process.

D - Incorrect: Backup restores a past copy instead of directly reversing this transaction.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB30

> A checkpoint is neither a full backup nor an automatic commit of active transactions.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

> • Recovery from catastrophic
> • If there is extensive damage to the wide portion of the database

• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 207

Question 97 | AM0208

What is the purpose of a checkpoint in the recovery process?

A. To save the current state of the database

B. To recover from a system crash

C. To log all transaction operations

D. To enforce data integrity constraints

Hint

Identify which history a recovery boundary can safely exclude and whether active work spans it.

Answer A: To save the current state of the database

A checkpoint synchronizes database and log by forcing modified buffers and recording the checkpoint (pp.105,107), saving a durable recovery boundary and reducing later work. It is not necessarily a full snapshot copy.

Why each choice is right or wrong

A - Correct: Saving synchronized current state best matches the listed procedure.

B - Incorrect: Crash recovery uses checkpoints, but this is a later use rather than checkpoint creation itself.

C - Incorrect: Logging every operation is the transaction log's role, not checkpoint's distinguishing purpose.

D - Incorrect: Constraint enforcement is not the checkpoint procedure.

Primary note proof

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history. In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo. Active transactions can span the checkpoint, so their relevant earlier updates can still need undo. Do not assume every transaction before a checkpoint needs no processing.

Study-note section ADB07

> Insert and delete also modify data; sorting is a query-processing operation rather than a primitive item update.

Study-note section ADB07

> Real systems may use physical or logical records, and need not log every read.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 105 | lines 25-36

Check pointing
• Log file is used to recover failed DB but we may not know how far back in the log to search. Thus
• Checkpoint is a Point of synchronization between database and log file.
• Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.
• When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 208

Question 98 | AM0209

In a log-based crash restart that uses durable checkpoint state plus relevant transaction logs, which listed labels describe the checkpoint-assisted technique and its overall restoration process? Select all that apply.

A. Rollback recovery

B. Checkpoint recovery

C. Restart recovery

D. Backup recovery

Hint

Identify which history a recovery boundary can safely exclude and whether active work spans it.

Answer B, C: B. Checkpoint recovery; C. Restart recovery

Checkpoint-assisted recovery uses recorded recovery state to limit the necessary log history. Restart recovery is the overall process that restores a valid state after the crash while preserving durable commits and removing incomplete effects. Both labels describe the specified procedure at different levels. Rollback names only undo work, and backup recovery would require starting from a separate stored copy.

Why each choice is right or wrong

A - Incorrect: Rollback specifically undoes incomplete effects rather than naming the checkpoint-plus-log restart procedure.

B - Correct: Checkpoint-assisted recovery uses durable checkpoint information with the relevant log history.

C - Correct: Restart recovery can use checkpoints and transaction logs to reconstruct valid state.

D - Incorrect: Backup recovery starts from a separate database copy, which the description does not mention.

Primary note proof

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history. In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo. Active transactions can span the checkpoint, so their relevant earlier updates can still need undo. Do not assume every transaction before a checkpoint needs no processing.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Study-note section ADB26

> Rollback handles failed/uncommitted work; redo preserves committed work missing from pages; checkpoints reduce log-search work; backups provide another stored copy.

Related reading (context, not primary note proof)

Study-note section ADB27

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 105 | lines 25-36

> Check pointing
> • Log file is used to recover failed DB but we may not know how far back in the

log to search. Thus
• Checkpoint is a Point of synchronization between database and log file.
• Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.
• When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 209

Question 99 | AM0210

Which overall crash-recovery process preserves durable committed work and removes incomplete effects after a system failure, leaving any business-level retry of formerly active transactions to a new controlled attempt?

A. Rollback recovery

B. Checkpoint recovery

C. Restart recovery

D. Backup recovery

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer C: C. Restart recovery

Restart recovery restores a valid database state after a crash. It preserves durable committed effects and removes incomplete effects using the required recovery mechanisms. Transactions active at failure are not automatically rerun and committed; a business retry is a separate controlled attempt. Rollback and checkpoint processing can be components, while backup recovery starts from another stored copy when that is needed.

Why each choice is right or wrong

A - Incorrect: Rollback names undo of incomplete work rather than the entire restart process.

B - Incorrect: Checkpoint information guides recovery rather than naming its whole process.

C - Correct: Restart recovery is the overall crash-restoration process described.

D - Incorrect: Backup restoration starts from a separate copy, which this scenario does not require.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Study-note section ADB26

> Rollback handles failed/uncommitted work; redo preserves committed work missing from pages; checkpoints reduce log-search work; backups provide another stored copy.

Related reading (context, not primary note proof)

Study-note section ADB32

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 105 | lines 25-36

> Check pointing
> • Log file is used to recover failed DB but we may not know how far back in the log to search. Thus
> • Checkpoint is a Point of synchronization between database and log file.
> • Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.

• When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Fundamental of Database Systems module | PDF page 108 | lines 24-29

• During recovery, all transactions of the commit table are redone and all transactions of active tables are ignored since none of their AFIMs reached the database.
• It is possible that a commit table transaction may be redone twice but this does not create any inconsistency because of a redone is —idempotent‖,
• that is, one redone for an AFIM is equivalent to multiple redone for the same AFIM.

Fundamental of Database Systems module | PDF page 110 | lines 14-30

◦ Deferred Update:
• All transaction updates are recorded in the local workspace (cache)
• All modified data items in the cache is then written after a transaction ends its execution or after a fixed number of transactions have completed their execution.
• During commit the updates are first recorded on the log and then on the database
• If a transaction fails before reaching its commit point undo is not needed because it didn‘t change the database yet
• If a transaction fails after commit (writing on the log) but before finishing saving to the data base redoing is needed from the log
◦ Immediate Update:
• As soon as a data item is modified in cache, the disk copy is updated.
• These update are first recorded on the log and on the disk by force writing, before the database is updated
• If a transaction fails after recording some change to the database but before reaching its commit point , this will be rolled back

Fundamental of Database Systems module | PDF page 110 | lines 5-13

◦ Repeating history during redo:
• ARIES will retrace all actions of the database system prior to the crash to reconstruct the database state when the crash occurred.
• Transaction that were un committed at the time of crash are undone
◦ Logging changes during undo:
• It will prevent ARIES from repeating the completed undo operations if a failure occurs during recovery, which causes a restart of the recovery process.
Data Update : Four types

Question sources

Computer Science MCQ practice collection · Question 210

Question 100 | AM0212

After a power failure loses volatile memory but leaves persistent storage intact, which term names the overall process that restores a valid database while preserving durable commits and removing incomplete effects? Choose the umbrella process rather than one component.

A. Rollback recovery

B. Checkpoint recovery

C. Restart recovery

D. Backup recovery

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer C: C. Restart recovery

Restart recovery is the overall restoration process following a system crash or power loss when persistent storage survives. It preserves durable commits and removes incomplete effects using the appropriate logged recovery work. Rollback may supply undo and checkpoints can shorten log searches, but these are cooperating mechanisms within the process. Backup restoration addresses a separate stored copy and is not required by this stated failure.

Why each choice is right or wrong

A - Incorrect: Rollback is an undo component rather than the overall process.

B - Incorrect: Checkpoint-assisted work guides restart rather than naming its umbrella process.

C - Correct: Restart recovery names the complete restoration after this system crash.

D - Incorrect: The stated intact persistent storage does not call for restoring a separate backup.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Study-note section ADB30

Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Related reading (context, not primary note proof)

Study-note section ADB27

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 212

Question 101 | AM0213

In a conventional transaction-log system, which listed mechanisms use logged updates or checkpoint/log state for undo or restart? Select all that apply. Backup recovery in D means restoring only a standalone full backup, without replaying archived logs.

A. Rollback recovery

B. Checkpoint recovery

C. Restart recovery

D. Backup recovery

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer A, B, C: A. Rollback recovery; B. Checkpoint recovery; C. Restart recovery

Logged rollback uses before-images to remove incomplete effects. Checkpoint-assisted recovery combines durable recovery state with relevant log history, and log-based restart uses that history to preserve commits and remove incomplete work. All three fit the specified model. A standalone backup restore instead copies a separately stored database image; the question excludes archived-log replay from D so that this alternative does not overlap.

Why each choice is right or wrong

A - Correct: The logged rollback uses before-images for undo.

B - Correct: Checkpoint-assisted recovery uses recovery-state records and relevant log history.

C - Correct: The specified log-based restart uses durable transaction history.

D - Incorrect: A standalone image restore without log replay does not use the specified logged recovery mechanism.

Primary note proof

Study-note section ADB27

> Write-ahead logging requires the update log needed for recovery to reach stable storage before the corresponding changed data page reaches disk. To acknowledge a durable commit, force the transaction's required log records, including its commit record, to stable storage. A buffer can be changed before that log flush; WAL orders durable writes, not every CPU assignment.

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward.

Study-note section ADB30

Active transactions can span the checkpoint, so their relevant earlier updates can still need undo.

Related reading (context, not primary note proof)

Study-note section ADB26

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 103 | lines 6-23

• Recovery from catastrophic
• If there is extensive damage to the wide portion of the database
• This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 104 | lines 19-25

• For Undo: Before a data item‘s AFIM is flushed to the database disk (overwriting the BFIM) its BFIM must be written to the log and the log must be saved on a stable store (log disk).
• For Redo: Before a transaction executes its commit operation, all its AFIMs must be written to the log and the log must be saved on a stable store.
• Standard Recovery Terminology

Fundamental of Database Systems module | PDF page 105 | lines 25-36

Check pointing
• Log file is used to recover failed DB but we may not know how far back in the log to search. Thus
• Checkpoint is a Point of synchronization between database and log file.
• Time to time (randomly or under some criteria) the database flushes its buffer to database disk to minimize the task of recovery.
• When failure occurs, redo all transactions that committed since the checkpoint

and undo all transactions active at time of crash.
• In previous example ,Figure 2, on slide18 , with checkpoint at time tc, changes made by T2 and T3 have been written to secondary storage. Thus: only redo T4 and T5, undo transactions T1 and T6.
• At what interval to take a check pointing may be measured :

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science MCQ practice collection · Question 213

Question 102 | AM0214

Which recovery method is used to recover from media failures, such as disk crashes?

A. Rollback recovery

B. Checkpoint recovery

C. Restart recovery

D. Backup recovery

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer D: Backup recovery

For disk media destruction, restore a backup copy and reconstruct committed operations from the backup log (p.103). D supplies the missing database contents.

Why each choice is right or wrong

A - Incorrect: Undo alone does not recreate a destroyed database copy.

B - Incorrect: Checkpoint synchronization alone is not an independent media backup.

C - Incorrect: A restart without restoring lost storage is insufficient.

D - Correct: Backup recovery is the required catastrophic/media restoration approach.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB26

> Rollback handles failed/uncommitted work; redo preserves committed work missing from pages; checkpoints reduce log-search work; backups provide another stored copy.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 97 | lines 4-13

> i. Purpose of Database Recovery
> ◦ To bring the database into the last consistent state, which existed prior to the failure.
> ◦ To preserve transaction properties (Atomicity &and Durability).
> • The recovery manager of a DBMS is responsible to ensure
> • atomicity by undoing the action of transaction that do not commit
> • Durability by making sure that all actions of committed transaction survive system crash
> ii. Types of Failure
> ◦ The database may become unavailable for use due to :

Fundamental of Database Systems module | PDF page 103 | lines 6-23

> • Recovery from catastrophic
> • If there is extensive damage to the wide portion of the database
> • This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure

• Recovery from non-catastrophic failure
• When the database is not physically damaged but has be come in consistent
• The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
• For instance,
• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Question sources

Computer Science MCQ practice collection · Question 214

Question 103 | AM0215

What is a distributed database system?

A. A database system that is spread across multiple locations

B. A database system that uses multiple database management systems

C. A database system that supports multiple database schemas

D. A database system that is used for storing distributed data

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer A: A database system that is spread across multiple locations

A DDB is multiple logically related databases distributed over a computer network, managed transparently (p.114). A captures the physical distribution; logical relatedness and coordinated management are implicit qualifications.

Why each choice is right or wrong

A - Correct: Multiple locations captures the defining distribution.

B - Incorrect: Different DBMS products are not required; homogeneous DDBs exist.

C - Incorrect: Multiple schemas are not the physical distribution definition.

D - Incorrect: Storing distributed data is circular and omits the network/location criterion.

Primary note proof

Study-note section ADB33

A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB33

Distribution adds communication, coordination and security costs; a site outage can still make needed data unavailable if no usable replica exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 114 | lines 3-13

6.1 Distributed Database Concepts
• A transaction can be executed by multiple networked computers in a unified manner.
• A distributed database (DDB) processes Unit of execution (a transaction) in a distributed manner.
• A distributed database (DDB) can be defined as :
- A collection of multiple logically related database distributed over a computer network, and a distributed database management system as a software system that manages a distributed database while making the distribution transparent to the user.
- The physical placement of data (files, relations, etc.) which is not known to the user (distribution transparency).

Variant 1 | Alternative version 1

What is a distributed database system?

A. A system that uses multiple database management systems

B. A system that supports multiple database schemas

C. A system that spans multiple locations or sites

D. A system that replicates data across multiple databases

Answer C: A system that spans multiple locations or sites

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Incorrect: Multiple DBMS products are not required for distribution.

B - Incorrect: Multiple schemas are not the spatial/network definition.

C - Correct: Spanning sites matches the distributed placement definition on p.114.

D - Incorrect: Replication is optional and narrower than a distributed database.

Primary note proof

Study-note section ADB33

> A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB33

> Distribution adds communication, coordination and security costs; a site outage can still make needed data unavailable if no usable replica exists.

Question sources

Computer Science MCQ practice collection · Question 215

Computer Science MCQ practice collection · Question 225

Question 104 | AM0216

What is the key advantage of a distributed database system?

A. Improved data security

B. Reduced data redundancy

C. Enhanced data availability

D. Simplified data querying

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer C: Enhanced data availability

Multiple nodes allow others to do the job if one fails (p.116), improving availability. The module lists integrity/security difficulty and complexity among disadvantages (p.117).

Why each choice is right or wrong

A - Incorrect: Improved security is not guaranteed; distributed access complicates control.

B - Incorrect: Replication stores multiple copies rather than necessarily reducing redundancy.

C - Correct: Availability is explicitly improved through multiple nodes.

D - Incorrect: Querying across sites adds network and optimization complexity rather than inherently simplifying it.

Primary note proof

Study-note section ADB33

A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB37

Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Study-note section ADB33

Distribution adds communication, coordination and security costs; a site outage can still make needed data unavailable if no usable replica exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 116 | lines 2-28

• Advantages of DDB :
i. Distribution and Network transparency:
• Users do not have to worry about operational details of the network.
» There is Location transparency, which refers to freedom of issuing command from any location without affecting its working.
» Then there is Naming transparency, which allows access to any names object (files, relations, etc.) from any location.
ii. Replication transparency:
• It allows to store copies of a data at multiple sites as shown in the above diagram.
• This is done to minimize access time to the required data.
iii. Fragmentation transparency:
• Allows to fragment a relation horizontally (create a subset of rows of a relation) or vertically (create a subset of columns of a relation).
iv. Increased reliability and availability:
• Reliability refers to system live time, that is, system is running efficiently most of the time. Availability is the probability that the system is continuously available (usable or accessible) during a time interval.
• A distributed database system has multiple nodes (computers) and if one fails then others are available to do the job.
v. Improved performance:
• A distributed DBMS fragments the database to keep data closer to where it is needed most.
• This reduces data management (access and modification) time significantly.

Fundamental of Database Systems module | PDF page 117 | lines 15-31

6.2 Data Replication and Fragmentation: Distributed data storage
• There are two approaches to store the relation in the distributed database : Replication and Fragmentation
I. Data Replication
• The system maintain several identical copies of the relation & store each copy at a different site
• In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead
II. Data Fragmentation
- Split a relation into logically related and correct parts.
- The main reasons for fragmenting a relation are
• Efficiency- data that is not needed by the local applications is not stored
• Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency
- but reconstruction of the whole relation will require accessing data from all sites containing part of the relation

Question sources

Computer Science MCQ practice collection · Question 216

Question 105 | AM0217

What is a homogenous distributed database system?

A. A system that uses the same database management system across all sites

B. A system that stores the same type of data across all sites

C. A system that uses a single database schema across all sites

D. A system that shares the same network infrastructure across all sites

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer A: A system that uses the same database management system across all sites

A homogeneous distributed database uses the same or compatible DBMS technology at its sites. Sharing data types, a schema or network infrastructure does not by itself establish that DBMS property. In particular, identical DBMS technology does not force all local schemas to be identical, and compatible structures alone do not prove identical DBMSs. The definition concerns the database-management technology, so A identifies the relevant criterion without adding unnecessary sameness requirements.

Why each choice is right or wrong

A - Correct: Correct: the same DBMS technology across sites meets the homogeneous-system criterion.

B - Incorrect: Incorrect: matching data types alone does not establish that the sites use the same DBMS technology.

C - Incorrect: Incorrect: schema agreement alone does not establish DBMS homogeneity, and same DBMS technology need not imply identical local schemas.

D - Incorrect: Incorrect: shared networking does not establish the database-management technology used at the sites.

Primary note proof

Study-note section ADB33

> A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Related reading (context, not primary note proof)

Study-note section ADB34

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 118 | lines 27-32

> 6.3 Types of Distributed Database Systems
> • Homogeneous
> - All sites of the database system have identical setup, i.e., same database system software.
> - The system may have little or no local autonomy
> - The underlying operating systems can be a mixture of Linux, Window, Unix, etc.

Variant 1 | Alternative version 1

What is a homogenous distributed database system?

A. A system that uses the same database management system across all sites

B. A system that stores the same type of data across all sites

C. A system that uses a single database schema across all sites

D. A system that uses the same network infrastructure across all sites

Answer A: A system that uses the same database management system across all sites

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Correct: Identical DBMS software matches p.118.

B - Incorrect: Identical data types alone do not establish software homogeneity.

C - Incorrect: A schema is not the software identity criterion.

D - Incorrect: Network infrastructure is not the DBMS software criterion.

Primary note proof

Study-note section ADB33

> A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB34

> A tightly integrated distributed system can present one global schema and shared control.

Study-note section ADB34

> State how much schema, transaction and administrative autonomy local sites retain.

Question sources

Computer Science MCQ practice collection · Question 217

Computer Science MCQ practice collection · Question 226

Question 106 | AM0218

What is a heterogeneous distributed database system?

A. A system that uses different database management systems across all sites

B. A system that stores different types of data across all sites

C. A system that uses different database schemas across all sites

D. A system that uses different network infrastructures across all sites

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer A: A system that uses different database management systems across all sites

Heterogeneous means at least one database is from a different vendor (p.119), supporting different DBMS software. It does not require every pair of sites to differ; A’s all sites must be read as across the participating sites, not pairwise uniqueness.

Why each choice is right or wrong

A - Correct: Different database-system vendors/software captures the defining distinction with the stated qualification.

B - Incorrect: Different content/types alone do not establish vendor heterogeneity.

C - Incorrect: Federated systems may have one conceptual schema despite heterogeneity.

D - Incorrect: Network differences are not the database-vendor definition.

Primary note proof

Study-note section ADB33

A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB33

Heterogeneous design must reconcile naming, types, units, identifiers, query capabilities and transaction semantics.

Study-note section ADB33

ODBC and JDBC provide connectivity interfaces; they do not automatically solve semantic differences or create one distributed transaction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 119 | lines 11-20

• Heterogeneous
- At least one of the database must be from different vendor : two variants
- Federated: Each site may run different database system but the data access is managed through a single conceptual schema.
• This implies that the degree of local autonomy is minimum. Each site must adhere to a centralized access policy. There may be a global schema.
- Multidatabase: There is no one conceptual global schema. For data access a schema is constructed dynamically as needed by the application software.
Site 2
Site 3

Question sources

Computer Science MCQ practice collection · Question 218

Computer Science MCQ practice collection · Question 227

Question 107 | AM0219

What is data fragmentation in a distributed database system?

A. Storing data in multiple copies at different sites

B. Dividing data into smaller subsets and storing them at different sites

C. Merging data from different sites into a single copy

D. Replicating data across all sites in the system

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer B: Dividing data into smaller subsets and storing them at different sites

Fragmentation splits a relation into logically related correct parts (p.117); horizontal fragments select rows and vertical fragments select columns (p.118). B describes storing these subsets across sites.

Why each choice is right or wrong

A - Incorrect: Multiple identical copies describe replication.

B - Correct: Smaller subsets describe fragmentation.

C - Incorrect: Merging is reconstruction/integration, the opposite direction from splitting.

D - Incorrect: Copying data to all sites is replication rather than fragmentation.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Study-note section ADB24

> Encryption protects stored or transmitted content; integrity constraints restrict valid data; auditing records actions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 117 | lines 15-31

6.2 Data Replication and Fragmentation: Distributed data storage
• There are two approaches to store the relation in the distributed database : Replication and Fragmentation
I. Data Replication
• The system maintain several identical copies of the relation & store each copy at a different site
• In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead
II. Data Fragmentation
- Split a relation into logically related and correct parts.
- The main reasons for fragmenting a relation are
• Efficiency- data that is not needed by the local applications is not stored
• Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency
- but reconstruction of the whole relation will require accessing data from all sites containing part of the relation

Fundamental of Database Systems module | PDF page 118 | lines 27-32

6.3 Types of Distributed Database Systems
• Homogeneous
- All sites of the database system have identical setup, i.e., same database system software.
- The system may have little or no local autonomy
- The underlying operating systems can be a mixture of Linux, Window, Unix, etc.

Variant 1 | Alternative version 1

What is data fragmentation in a distributed database system?

A. Dividing data into smaller subsets and storing them at different sites

B. Replicating data across all sites for fault tolerance

C. Merging data from different sites into a single database

D. Encrypting data for secure transmission between sites

Answer A: Dividing data into smaller subsets and storing them at different sites

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Correct: Splitting into subsets matches fragmentation on pp.117-118.

B - Incorrect: Copies for availability describe replication.

C - Incorrect: Merging combines rather than splits the relations.

D - Incorrect: Encryption concerns protection rather than fragmentation.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Study-note section ADB24

> Encryption protects stored or transmitted content; integrity constraints restrict valid data; auditing records actions.

Question sources

Computer Science MCQ practice collection · Question 219

Computer Science MCQ practice collection · Question 228

Question 108 | AM0220

What is data replication in a distributed database system?

A. Storing data in multiple copies at different sites

B. Dividing data into smaller subsets and storing them at different sites

C. Merging data from different sites into a single copy

D. Deleting data that is no longer needed from the database

Hint

Separate copy placement, propagation timing and update-maintenance cost.

Answer A: Storing data in multiple copies at different sites

Replication maintains several identical copies of a relation at different sites (p.117). This differs from dividing it into fragments.

Why each choice is right or wrong

A - Correct: Multiple copies at different sites exactly matches.

B - Incorrect: Subsets describe fragmentation.

C - Incorrect: Merging into one copy is not maintaining multiple copies.

D - Incorrect: Deletion removes data rather than replicating it.

Primary note proof

Study-note section ADB36

Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB37

Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Study-note section ADB24

Encryption protects stored or transmitted content; integrity constraints restrict valid data; auditing records actions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 117 | lines 15-31

6.2 Data Replication and Fragmentation: Distributed data storage
• There are two approaches to store the relation in the distributed database : Replication and Fragmentation
I. Data Replication
• The system maintain several identical copies of the relation & store each copy at a different site
• In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead
II. Data Fragmentation
- Split a relation into logically related and correct parts.
- The main reasons for fragmenting a relation are
• Efficiency- data that is not needed by the local applications is not stored
• Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency
- but reconstruction of the whole relation will require accessing data from all sites containing part of the relation

Variant 1 | Alternative version 1

What is data replication in a distributed database system?

A. Storing multiple copies of data at different sites

B. Dividing data into smaller subsets for efficient storage

C. Merging data from different sites into a single database

D. Encrypting data to ensure security during transmission

Answer A: Storing multiple copies of data at different sites

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Correct: Identical copies at different sites match p.117.

B - Incorrect: Subsets describe fragmentation, not replication.

C - Incorrect: Merging to one database does not retain distributed copies.

D - Incorrect: Encryption does not create replicated copies.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Study-note section ADB24

> Encryption protects stored or transmitted content; integrity constraints restrict valid data; auditing records actions.

Question sources

Computer Science MCQ practice collection · Question 220

Computer Science MCQ practice collection · Question 229

Question 109 | AM0221

Which transparency hides the physical site storing a named database object, so users can refer to that object without specifying its site?

A. Local-only data access

B. Fragmentation transparency

C. Location transparency

D. Replication transparency

Hint

Identify exactly which detail is hidden from the user.

Answer C: Location transparency

Location transparency hides the physical site holding a named logical object. Fragmentation transparency instead hides the existence and reconstruction of fragments, while replication transparency hides which copy is used and how copies are maintained. Those properties can coexist in a distributed database, so a vague single-logical-database experience would not identify only one. The revised stem isolates hidden site placement. Local-only access restricts what can be accessed rather than making remote placement transparent.

Why each choice is right or wrong

A - Incorrect: Incorrect: restricting access to local data does not hide the placement of a named distributed object.

B - Incorrect: Incorrect: fragmentation transparency hides splitting and reconstruction, not specifically the object's physical site.

C - Correct: Correct: location transparency hides which physical site stores the named object.

D - Incorrect: Incorrect: replication transparency hides copy selection and maintenance, not specifically site placement.

Primary note proof

Study-note section ADB35

> Location transparency hides which site holds a named object. Fragmentation transparency hides the existence and reconstruction of fragments: users query the whole logical relation. Replication transparency hides which copy is used and how copies are maintained. The broad single-logical-database experience may involve all three.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 114 | lines 3-13

> 6.1 Distributed Database Concepts
> • A transaction can be executed by multiple networked computers in a unified manner.
> • A distributed database (DDB) processes Unit of execution (a transaction) in a distributed manner.
> • A distributed database (DDB) can be defined as :
> - A collection of multiple logically related database distributed over a computer network, and a distributed database management system as a software system that manages a distributed database while making the distribution transparent to the user.
> - The physical placement of data (files, relations, etc.) which is not known to the user (distribution transparency).

Fundamental of Database Systems module | PDF page 116 | lines 2-28

> • Advantages of DDB :
> i. Distribution and Network transparency:
> • Users do not have to worry about operational details of the

network.
» There is Location transparency, which refers to freedom of issuing command from any location without affecting its working.
» Then there is Naming transparency, which allows access to any names object (files, relations, etc.) from any location.
ii. Replication transparency:
• It allows to store copies of a data at multiple sites as shown in the above diagram.
• This is done to minimize access time to the required data.
iii. Fragmentation transparency:
• Allows to fragment a relation horizontally (create a subset of rows of a relation) or vertically (create a subset of columns of a relation).
iv. Increased reliability and availability:
• Reliability refers to system live time, that is, system is running efficiently most of the time. Availability is the probability that the system is continuously available (usable or accessible) during a time interval.
• A distributed database system has multiple nodes (computers) and if one fails then others are available to do the job.
v. Improved performance:
• A distributed DBMS fragments the database to keep data closer to where it is needed most.
• This reduces data management (access and modification) time significantly.

Question sources

Computer Science MCQ practice collection · Question 221

Question 110 | AM0222

A user explicitly names a fragment and can query it without specifying or knowing the physical site that stores it. Which transparency is being provided? The existence of the fragment is visible.

A. Local-only data access

B. Fragmentation transparency

C. Location transparency

D. Replication transparency

Hint

Identify exactly which detail is hidden from the user.

Answer C: Location transparency

Location transparency hides the physical site storing a named object. The user in this scenario still names the fragment, so fragmentation remains visible; fragmentation transparency would hide the split and reconstruction. Replication transparency hides which copy is selected or maintained. Restricting users to local data is an access limitation rather than concealment of a remote fragment's placement. C therefore identifies the particular hidden detail without treating all transparency properties as interchangeable.

Why each choice is right or wrong

A - Incorrect: Incorrect: local-only access restricts accessible data rather than hiding a named remote fragment's physical site.

B - Incorrect: Incorrect: the user explicitly names the fragment, so its existence is not hidden.

C - Correct: Correct: the fragment's physical site is hidden while its logical name remains usable.

D - Incorrect: Incorrect: replica selection and maintenance are different hidden details from the specified site placement.

Primary note proof

Study-note section ADB35

> Location transparency hides which site holds a named object. Fragmentation transparency hides the existence and reconstruction of fragments: users query the whole logical relation. Replication transparency hides which copy is used and how copies are maintained. The broad single-logical-database experience may involve all three.

Study-note section ADB35

> If users explicitly name a fragment but need not specify its site, fragmentation is visible while location can remain transparent. Distinguish whether fragment names, storage sites or both are hidden from the user.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 114 | lines 3-13

> 6.1 Distributed Database Concepts
> • A transaction can be executed by multiple networked computers in a unified manner.
> • A distributed database (DDB) processes Unit of execution (a transaction) in a distributed manner.
> • A distributed database (DDB) can be defined as :
> - A collection of multiple logically related database distributed over a computer network, and a distributed database management system as a software system that manages a distributed database while making the distribution transparent to the user.
> - The physical placement of data (files, relations, etc.) which is not known to the user (distribution transparency).

Fundamental of Database Systems module | PDF page 116 | lines 2-28

• Advantages of DDB :
i. Distribution and Network transparency:
• Users do not have to worry about operational details of the network.
» There is Location transparency, which refers to freedom of issuing command from any location without affecting its working.
» Then there is Naming transparency, which allows access to any names object (files, relations, etc.) from any location.
ii. Replication transparency:
• It allows to store copies of a data at multiple sites as shown in the above diagram.
• This is done to minimize access time to the required data.
iii. Fragmentation transparency:
• Allows to fragment a relation horizontally (create a subset of rows of a relation) or vertically (create a subset of columns of a relation).
iv. Increased reliability and availability:
• Reliability refers to system live time, that is, system is running efficiently most of the time. Availability is the probability that the system is continuously available (usable or accessible) during a time interval.
• A distributed database system has multiple nodes (computers) and if one fails then others are available to do the job.
v. Improved performance:
• A distributed DBMS fragments the database to keep data closer to where it is needed most.
• This reduces data management (access and modification) time significantly.

Fundamental of Database Systems module | PDF page 118 | lines 27-32

6.3 Types of Distributed Database Systems
• Homogeneous
- All sites of the database system have identical setup, i.e., same database system software.
- The system may have little or no local autonomy
- The underlying operating systems can be a mixture of Linux, Window, Unix, etc.

Question sources

Computer Science MCQ practice collection · Question 222

Question 111 | AM0223

Which data transparency level ensures that users are not aware of the presence of multiple copies of data in different sites?

A. Local data transparency

B. Fragmentation transparency

C. Location transparency

D. Replication transparency

Hint

Identify exactly which detail is hidden from the user.

Answer D: Replication transparency

The module combines transparent distribution (p.114) with replication as storing copies at multiple sites (pp.116-117). Hiding this copying is replication transparency.

Why each choice is right or wrong

A - Incorrect: Local transparency does not identify multiple copies.

B - Incorrect: Fragmentation concerns subsets rather than duplicates.

C - Incorrect: Location hides placement but does not specifically name the copying aspect.

D - Correct: Replication transparency is the copy-specific aspect.

Primary note proof

Study-note section ADB35

> Location transparency hides which site holds a named object. Fragmentation transparency hides the existence and reconstruction of fragments: users query the whole logical relation. Replication transparency hides which copy is used and how copies are maintained. The broad single-logical-database experience may involve all three.

Study-note section ADB35

> If users explicitly name a fragment but need not specify its site, fragmentation is visible while location can remain transparent.

Study-note section ADB35

> Column subsets define vertical fragmentation, not a type of transparency.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 114 | lines 3-13

> 6.1 Distributed Database Concepts
> • A transaction can be executed by multiple networked computers in a unified manner.
> • A distributed database (DDB) processes Unit of execution (a transaction) in a

distributed manner.
• A distributed database (DDB) can be defined as :
- A collection of multiple logically related database distributed over a computer network, and a distributed database management system as a software system that manages a distributed database while making the distribution transparent to the user.
- The physical placement of data (files, relations, etc.) which is not known to the user (distribution transparency).

Fundamental of Database Systems module | PDF page 116 | lines 2-28

• Advantages of DDB :
i. Distribution and Network transparency:
• Users do not have to worry about operational details of the network.
» There is Location transparency, which refers to freedom of issuing command from any location without affecting its working.
» Then there is Naming transparency, which allows access to any names object (files, relations, etc.) from any location.
ii. Replication transparency:
• It allows to store copies of a data at multiple sites as shown in the above diagram.
• This is done to minimize access time to the required data.
iii. Fragmentation transparency:
• Allows to fragment a relation horizontally (create a subset of rows of a relation) or vertically (create a subset of columns of a relation).
iv. Increased reliability and availability:
• Reliability refers to system live time, that is, system is running efficiently most of the time. Availability is the probability that the system is continuously available (usable or accessible) during a time interval.
• A distributed database system has multiple nodes (computers) and if one fails then others are available to do the job.
v. Improved performance:
• A distributed DBMS fragments the database to keep data closer to where it is needed most.
• This reduces data management (access and modification) time significantly.

Fundamental of Database Systems module | PDF page 117 | lines 15-31

6.2 Data Replication and Fragmentation: Distributed data storage
• There are two approaches to store the relation in the distributed database : Replication and Fragmentation
I. Data Replication
• The system maintain several identical copies of the relation & store each copy at a different site
• In general it enhance the performance of read operation and increase the

availability of data to read only transaction. However, update transactions incur greater overhead
II. Data Fragmentation
- Split a relation into logically related and correct parts.
- The main reasons for fragmenting a relation are
• Efficiency- data that is not needed by the local applications is not stored
• Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency
- but reconstruction of the whole relation will require accessing data from all sites containing part of the relation

Question sources

Computer Science MCQ practice collection · Question 223

Question 112 | AM0224

What is the role of a distributed query processor in a distributed database system?

A. It ensures the consistency of data across all sites.

B. It manages the distribution and retrieval of data across multiple sites.

C. It controls the access permissions of users in the system.

D. It handles the physical storage of data at each site.

Hint

Compare local processing with the bytes and messages moved between sites.

Answer B: It manages the distribution and retrieval of data across multiple sites.

Distributed query processing compares moving source relations and results among sites, selecting the lower-transfer strategy (pp.120-121). B best captures distributed retrieval, understood as planning transfers for a query rather than deciding the physical database design.

Why each choice is right or wrong

A - Incorrect: Global consistency is assigned to concurrency control/recovery rather than query transfer planning.

B - Correct: Managing query data transfer and retrieval matches the worked example.

C - Incorrect: Access permissions are authorization, not the worked query role.

D - Incorrect: Physical local storage management is not choosing distributed query strategies.

Primary note proof

Study-note section ADB38

Distributed processing decomposes a logical query, localizes referenced relations to fragments/replicas, chooses global placement/order and optimizes local subplans. Include bytes

moved, message latency, bandwidth and local work. Apply filters and projections before shipping when equivalent. A semijoin retains only rows of one input that have a match in the other, returning that input's attributes.

Study-note section ADB24

Authentication establishes identity; authorization/access control determines permitted actions.

Study-note section ADB34

A tightly integrated distributed system can present one global schema and shared control.

Related reading (context, not primary note proof)

Study-note section ADB39

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 120 | lines 16-33

6.4 Query Processing in Distributed Databases
• Issues
- Cost of transferring data (files and results) over the network.
• This cost is usually high, so some optimization is necessary.
• Example: suppose there are three sites. Where the relation Employee at site 1, Department at Site 2 and no relation at site 3
- Employee at site 1. 10,000 rows. Row size = 100 bytes. Table size = 106 bytes.
- Department at Site 2. 100 rows. Row size = 35 bytes. Table size = 3,500 bytes.
- And a query is initiated from S3 to retrieve employees [First Name (15 byte long), Last name (15 byte long) and Department name (10 byte long) total of 40 bytes]
• Q: For each employee, retrieve employee Fname, Lname, and department name
• Q: δFname,Lname,Dname (Employee Dno = Dnumber Department)
• Assumption
Site 3

Fundamental of Database Systems module | PDF page 121 | lines 8-24

• Strategies : Minimizing data transfer.
- Transfer Employee and Department to site 3.
• Total transfer bytes = 1,000,000 + 3500 = 1,003,500 bytes.
- Transfer Employee to site 2, execute join at site 2 and send the result to site 3.
• Transferring employees data from site 1 to site 2: 1,000,000 bytes
• Query result size = 40 * 10,000 = 400,000 bytes.
• Total transfer size = 1,000,000 + 400,000 = 1,400,000 bytes.
- Transfer Department relation to site 1, execute the join at site 1, and send the result to site 3.
• Data Transfer from site 2 to site 1: 3500 bytes

• Query result size = 40 * 10,000 = 400,000 bytes
• Total bytes transferred = 3500+ 400,000 = 403,500 bytes.
• Preferred approach: strategy 3.
Example 2 : Consider the query
- Q‟: For each department, retrieve the department name ,Fname and LName of the department manager
• Relational Algebra expression:

Question sources

Computer Science MCQ practice collection · Question 224

Question 113 | AM0230

Which condition specifically makes a distributed database environment homogeneous, rather than merely being useful for distributed operation?

A. Consistency of database schemas across all sites

B. Compatibility of network infrastructures across all sites

C. Standardization of database management systems across all sites

D. Replication of data at each site for fault tolerance

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer C: Standardization of database management systems across all sites

Homogeneity is defined by using the same or compatible DBMS technology across sites. Schema compatibility, functioning network connections and replication can be useful design concerns, but none alone establishes that common DBMS technology. A homogeneous system may have different local schemas, and replication is not required merely because data is distributed. Narrowing the question to the defining condition preserves C while acknowledging that the other design concerns can matter in practice.

Why each choice is right or wrong

A - Incorrect: Incorrect: schema compatibility can be useful but does not establish common DBMS technology.

B - Incorrect: Incorrect: compatible networking supports communication but does not establish DBMS homogeneity.

C - Correct: Correct: DBMS standardization supplies the defining common technology across sites.

D - Incorrect: Incorrect: replication can improve availability but is optional and does not define homogeneity.

Primary note proof

Study-note section ADB33

A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB37

Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 118 | lines 27-32

6.3 Types of Distributed Database Systems
• Homogeneous
- All sites of the database system have identical setup, i.e., same database system software.
- The system may have little or no local autonomy
- The underlying operating systems can be a mixture of Linux, Window, Unix, etc.

Question sources

Computer Science MCQ practice collection · Question 230

Question 114 | AM0231

A heterogeneous distributed database retains different local DBMSs and schemas and must integrate them without requiring full replication. Which integration requirements apply? Select all that apply. D means coherent schema semantics through mappings, not identical schemas.

A. Replication of data at each site for fault tolerance

B. Standardization of database management systems across all sites

C. Compatibility of network infrastructures across all sites

D. Consistency of database schemas across all sites

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer C, D: C. Compatibility of network infrastructures across all sites; D. Consistency of database schemas across all sites

Integration requires interoperable communication so requests and results can move between sites. It also requires coherent mappings for names, types, identifiers and other schema meanings. These support C and D. Replacing every DBMS with one product would remove the retained heterogeneity, and full replication is an optional placement decision rather than a requirement for integrating different local systems.

Why each choice is right or wrong

A - Incorrect: Copying all data everywhere is not required for the stated integration.

B - Incorrect: The scenario retains differing DBMSs rather than replacing all with one standard.

C - Correct: Interoperable communication is needed to exchange requests and results.

D - Correct: Mappings must reconcile schema meanings without requiring identical local schemas.

Primary note proof

Study-note section ADB33

> A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB33

> Heterogeneous design must reconcile naming, types, units, identifiers, query capabilities and transaction semantics.

Study-note section ADB37

> Full replication does not inherently mean every copy is updated simultaneously; its synchronization rule must be stated.

Related reading (context, not primary note proof)

Study-note section ADB40

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 119 | lines 11-20

> • Heterogeneous
> - At least one of the database must be from different vendor : two variants
> - Federated: Each site may run different database system but the data access is managed through a single conceptual schema.
> • This implies that the degree of local autonomy is minimum. Each site must adhere to a centralized access policy. There may be a global schema.
> - Multidatabase: There is no one conceptual global schema. For data access a schema is constructed dynamically as needed by the application software.
> Site 2
> Site 3

Fundamental of Database Systems module | PDF page 117 | lines 15-31

> 6.2 Data Replication and Fragmentation: Distributed data storage
> • There are two approaches to store the relation in the distributed database : Replication

and Fragmentation
I. Data Replication
• The system maintain several identical copies of the relation & store each copy at a different site
• In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead
II. Data Fragmentation
- Split a relation into logically related and correct parts.
- The main reasons for fragmenting a relation are
• Efficiency- data that is not needed by the local applications is not stored
• Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency
- but reconstruction of the whole relation will require accessing data from all sites containing part of the relation

Question sources

Computer Science MCQ practice collection · Question 231

Question 115 | AM0232

Which approach is commonly used for query processing in a distributed database system?

A. Centralized query processing

B. Distributed query processing

C. Parallel query processing

D. Sequential query processing

Hint

Compare local processing with the bytes and messages moved between sites.

Answer B: Distributed query processing

The module's distributed-query section compares execution and data-transfer locations among multiple sites (pp.120-121). B names the overall approach. A chosen plan may execute a particular join at one site; that does not make the whole system centralized.

Why each choice is right or wrong

A - Incorrect: A local execution site can be part of a distributed plan, but centralized processing does not name the chapter's overall approach.

B - Correct: Distributed query processing matches the section and cross-site strategies.

C - Incorrect: Parallelism may be used, but simultaneous execution is not required by these examples.

D - Incorrect: A plan may contain sequential operations; this does not identify cross-site processing.

Primary note proof

Study-note section ADB38

Distributed processing decomposes a logical query, localizes referenced relations to fragments/replicas, chooses global placement/order and optimizes local subplans. Include bytes moved, message latency, bandwidth and local work. Apply filters and projections before shipping when equivalent. A semijoin retains only rows of one input that have a match in the other, returning that input's attributes.

Study-note section ADB38

If nearly all S matches, key shipping may not help.

Study-note section ADB22

Plan caching reuses an execution strategy, not the rows returned.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 120 | lines 16-33

6.4 Query Processing in Distributed Databases
• Issues
- Cost of transferring data (files and results) over the network.
• This cost is usually high, so some optimization is necessary.
• Example: suppose there are three sites. Where the relation Employee at
site 1, Department at Site 2 and no relation at site 3
- Employee at site 1. 10,000 rows. Row size = 100 bytes. Table size
= 106
bytes.
- Department at Site 2. 100 rows. Row size = 35 bytes. Table size =
3,500 bytes.
- And a query is initiated from S3 to retrieve employees [First Name (15 byte long),
Last name (15 byte long) and Department name (10 byte long) total of 40 bytes]
• Q: For each employee, retrieve employee Fname, Lname, and
department name
• Q: δFname,Lname,Dname (Employee Dno = Dnumber Department)
• Assumption
Site 3

Fundamental of Database Systems module | PDF page 121 | lines 8-24

• Strategies : Minimizing data transfer.
- Transfer Employee and Department to site 3.
• Total transfer bytes = 1,000,000 + 3500 = 1,003,500 bytes.
- Transfer Employee to site 2, execute join at site 2 and send the result to site 3.
• Transferring employees data from site 1 to site 2: 1,000,000 bytes

• Query result size = 40 * 10,000 = 400,000 bytes.
• Total transfer size = 1,000,000 + 400,000 = 1,400,000 bytes.
- Transfer Department relation to site 1, execute the join at site 1, and send the result to site 3.
• Data Transfer from site 2 to site 1: 3500 bytes
• Query result size = 40 * 10,000 = 400,000 bytes
• Total bytes transferred = 3500+ 400,000 = 403,500 bytes.
• Preferred approach: strategy 3.
Example 2 : Consider the query
- Q‟: For each department, retrieve the department name ,Fname and LName of the department manager
• Relational Algebra expression:

Question sources

Computer Science MCQ practice collection · Question 232

Question 116 | AM0233

What is the role of a distributed transaction coordinator in a distributed database system?

A. It ensures the consistency of data across all sites.

B. It manages the distribution and retrieval of data across multiple sites.

C. It controls the access permissions of users in the system.

D. It handles the physical storage of data at each site.

Hint

Distinguish coordinated transaction outcome from concurrent access and local storage.

Answer A: It ensures the consistency of data across all sites.

Global consistency is required of concurrency/recovery (p.122). Under the primary-site technique a coordinator manages transaction concurrency and commit (p.123). A is supported in that transaction-coordination sense, rather than as an unconditional guarantee against every error.

Why each choice is right or wrong

A - Correct: Coordination of concurrency and commit maintains transaction consistency across sites.

B - Incorrect: Query transfer/retrieval is a separate distributed-query task.

C - Incorrect: User permissions are authorization, not commit coordination.

D - Incorrect: Local physical storage is not the primary coordinator’s stated function.

Primary note proof

Study-note section ADB39

Two-phase commit (2PC) first asks participants to prepare and durably promise they can commit. If all vote yes, the coordinator records commit durably and tells participants; otherwise it decides abort. A prepared participant must retain enough state to obey the final decision. Coordinator/communication failure can leave it in doubt and block progress until the decision is learned. Timeout after preparation is not permission to invent a commit or abort.

Study-note section ADB39

The transaction coordinator coordinates decisions rather than physically storing every site's rows.

Study-note section ADB38

Distributed processing decomposes a logical query, localizes referenced relations to fragments/replicas, chooses global placement/order and optimizes local subplans.

Related reading (context, not primary note proof)

Study-note section ADB06

Study-note section ADB24

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 122 | lines 22-29

6.5 Concurrency Control and Recovery
• Distributed Databases encounter a number of concurrency control and recovery problems which are not present in centralized databases. Some of them are listed below.
• Dealing with multiple copies of data items:
• The concurrency control must maintain global consistency. Likewise the recovery mechanism must recover all copies and maintain consistency after recovery.
• Failure of individual sites:

Fundamental of Database Systems module | PDF page 123 | lines 15-29

• Distributed commit:
• A transaction may be fragmented and they may be executed by a number of sites. This require a two or three-phase commit approach for transaction commit.
• Distributed deadlock:
• Since transactions are processed at multiple sites, two or more sites may get involved in deadlock. This must be resolved in a distributed manner.
6.5.1 Distributed Concurrency control
i. Primary site technique: A single site is designated as a primary site which serves as a coordinator for transaction management.
• Transaction management:
- Concurrency control and commit are managed by this site.
- In two phase locking, this site manages locking and releasing data items. If all

transactions follow two-phase policy at all sites, then serializability is guaranteed.
- Advantages:

Question sources

Computer Science MCQ practice collection · Question 233

Question 117 | AM0234

Which listed approaches are replication policies or transfer methods in distributed databases, allowing dimensions to coexist? Select all that apply.

A. Eager replication

B. Lazy replication

C. Snapshot replication

D. Incremental replication

Hint

Separate copy placement, propagation timing and update-maintenance cost.

Answer A, B, C, D: A. Eager replication; B. Lazy replication; C. Snapshot replication; D. Incremental replication

Eager and lazy replication differ in when propagation occurs relative to the acknowledgment or commit point. Snapshot and incremental replication differ in whether a copy or changes are transferred. These are legitimate replication dimensions and can coexist, so all four qualify. The question asks for recognized approaches rather than claiming that one policy is most prevalent or universally preferable.

Why each choice is right or wrong

A - Correct: Eager replication coordinates propagation before the specified acknowledgment or commit point.

B - Correct: Lazy replication propagates changes later and can expose stale reads or require conflict handling.

C - Correct: Snapshot replication distributes a copy at intervals rather than necessarily sending each individual update.

D - Correct: Incremental replication transfers changes and can coexist with a propagation-timing policy.

Primary note proof

Study-note section ADB37

Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once. Copies

improve read locality and can improve availability but require storage and update propagation. They can also spread mistakes, so replication does not replace independent backups.

Study-note section ADB37

Eager/synchronous replication coordinates propagation before the specified commit/acknowledgment point, increasing update latency.

Study-note section ADB37

Snapshot replication transfers a copy at intervals; incremental replication transfers changes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 117 | lines 15-31

6.2 Data Replication and Fragmentation: Distributed data storage
• There are two approaches to store the relation in the distributed database : Replication and Fragmentation
I. Data Replication
• The system maintain several identical copies of the relation & store each copy at a different site
• In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead
II. Data Fragmentation
- Split a relation into logically related and correct parts.
- The main reasons for fragmenting a relation are
• Efficiency- data that is not needed by the local applications is not stored
• Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency
- but reconstruction of the whole relation will require accessing data from all sites containing part of the relation

Question sources

Computer Science MCQ practice collection · Question 234

Question 118 | AM0235

When comparing distributed query strategies, which factors can affect estimated execution cost or replica placement decisions? Select all that apply.

A. Cost of query execution

B. Size of the database

C. Network bandwidth

D. Data replication

Hint

Identify the resource objective, counts and unit costs before comparing execution strategies.

Answer A, B, C, D: A. Cost of query execution; B. Size of the database; C. Network bandwidth; D. Data replication

Execution cost summarizes the resources consumed by a candidate plan. Relation sizes affect scans and intermediate work, network bandwidth affects data-transfer time, and replica placement changes which copies are available near the computation. All these inputs can alter a distributed strategy's cost. They are complementary considerations rather than competing names for one exclusively important factor.

Why each choice is right or wrong

A - Correct: Estimated execution cost is the explicit basis for comparing candidate strategies.

B - Correct: Database and relation sizes affect scan work, intermediate sizes and algorithm feasibility.

C - Correct: Bandwidth affects transfer time when data or results move between sites.

D - Correct: Replica locations affect which copy can serve a query and its communication cost.

Primary note proof

Study-note section ADB19

> A cost model estimates resources for a candidate plan: page I/O, CPU, memory, temporary storage and network traffic. Catalog statistics include row/page counts, distinct values and distributions. Selection with fraction s predicts s×N rows. For non-NULL equijoin keys, a common estimate is |R|×|S|/max(V(R,k),V(S,k)), assuming uniform values and containment of the smaller distinct-value domain within the larger. Adjust counts for NULLs; if either non-NULL input is empty, the result is zero. Skew, correlations or incomplete domain overlap can invalidate this estimate; known disjoint domains give zero matches.

Study-note section ADB38

> Distributed processing decomposes a logical query, localizes referenced relations to fragments/replicas, chooses global placement/order and optimizes local subplans.

Study-note section ADB38

> Include bytes moved, message latency, bandwidth and local work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 41 | lines 25-29

> B. Cost Estimation Approach to Query Optimization
> - The main idea is to minimize he cost of processing a query. The cost function is comprised of:
> - I/O cost + CPU processing cost + communication cost + Storage cost
> - These components might have different weights in different processing environments

Fundamental of Database Systems module | PDF page 42 | lines 10-25

Cost Estimation for RA Operations
- Many different ways of implementing RA operations.
- Aim of QO is to choose most efficient one.
- Use formulae that estimate costs for a number of options, and select one with lowest cost.
- Consider only cost of disk access, which is usually dominant cost in QP.
- Many estimates are based on cardinality of the relation, so need to be able to estimate this.

Using Selectivity and Cost Estimates in Query Optimization
- Cost-based query optimization:
  - Estimate and compare the costs of executing a query using different execution strategies and choose the strategy with the lowest cost estimate. (Compare to heuristic query optimization)
- Issues
  - Cost function
  - Number of execution strategies to be considered
- Cost Components for Query Execution

Fundamental of Database Systems module | PDF page 120 | lines 16-33

6.4 Query Processing in Distributed Databases
• Issues
- Cost of transferring data (files and results) over the network.
• This cost is usually high, so some optimization is necessary.
• Example: suppose there are three sites. Where the relation Employee at site 1, Department at Site 2 and no relation at site 3
- Employee at site 1. 10,000 rows. Row size = 100 bytes. Table size = 106 bytes.
- Department at Site 2. 100 rows. Row size = 35 bytes. Table size = 3,500 bytes.
- And a query is initiated from S3 to retrieve employees [First Name (15 byte long), Last name (15 byte long) and Department name (10 byte long) total of 40 bytes]
• Q: For each employee, retrieve employee Fname, Lname, and department name
• Q: δFname,Lname,Dname (Employee Dno = Dnumber Department)
• Assumption
Site 3

Fundamental of Database Systems module | PDF page 42 | lines 15-17

- Many estimates are based on cardinality of the relation, so need to be able to estimate this.

Using Selectivity and Cost Estimates in Query Optimization

Question sources

Computer Science MCQ practice collection · Question 235

Question 119 | AM0236

What is the purpose of query optimization in a database system?

A. To ensure data consistency

B. To minimize query execution time

C. To maximize data replication

D. To manage database security

Hint

Separate query representation, strategy selection and execution.

Answer B: To minimize query execution time

Query optimization chooses an efficient execution strategy and minimizes resource usage including response time (p.27). Page 30 explicitly seeks faster queries and the most efficient sequence of steps.

Why each choice is right or wrong

A - Incorrect: Consistency is a transaction correctness concern, not the stated optimization objective.

B - Correct: Minimizing execution/response time captures the efficiency objective.

C - Incorrect: Maximizing replication is not the execution-plan goal.

D - Incorrect: Authorization/security is not query optimization's stated purpose.

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB17

> Query processing is the whole activity; selecting a suitable efficient strategy is optimization.

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work.

Related reading (context, not primary note proof)

Study-note section ADB19

Study-note section ADB24

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 27 | lines 4-15

What is Query Processing?
Steps required to transform high level SQL query into a correct and —efficient‖ strategy for execution and retrieval.
What is Query Optimization?
✓ The activity of choosing a single —efficient‖ execution strategy (from hundreds) as determined by database catalog statistics.
✓ The aim is to answer which relational algebra expression, equivalent to the given query, will lead to the most efficient solution plan?
✓ For each algebraic operator, what algorithm (of several available) do we use to compute that operator?
✓ How do operations pass data (main memory buffer, disk buffer,...)?
✓ Will this plan minimize resource usage? (CPU/Response Time/Disk)

Fundamental of Database Systems module | PDF page 30 | lines 17-30

1.8 Query Optimization
• Everyone wants the performance of their database to be optimal. In particular, there is often a requirement for a specific query or object that is query based, to run faster.
• Problem of query optimization is to find the sequence of steps that produces the answer to user request in the most efficient manner, given the database structure.
• The performance of a query is affected by the tables or views that underlies the query and by the complexity of the query.
• Given a request for data manipulation or retrieval, an optimizer will choose an optimal plan for evaluating the request from among the manifold alternative strategies. i.e. there are many ways (access paths) for accessing desired file/record.
• Hence ,DBMS is responsible to pick the best execution strategy based on various considerations(Least amount of I/O and CPU resources.)
• Example: Consider relations r(AB) and s(CD). We require r X s.
• Method 1 :

Question sources

Computer Science MCQ practice collection · Question 236

Question 120 | AM0237

What is a query plan in a database system?

A. A set of database schemas

B. A representation of the query result

C. A sequence of operations to execute a query

D. A collection of database indices

Hint

Separate query representation, strategy selection and execution.

Answer C: A sequence of operations to execute a query

An execution plan includes a relational-algebra query tree, access methods and operator implementations (p.44). Its operations run from leaves toward the root (p.29), so C captures the execution procedure, which can have a tree structure rather than a single linear list.

Why each choice is right or wrong

A - Incorrect: Schemas describe data structure rather than the executable operations.

B - Incorrect: The result is the output, whereas the plan specifies how to obtain it.

C - Correct: An operation sequence/tree with access methods is the stated plan concept.

D - Incorrect: Indices may be access methods within a plan but do not alone constitute it.

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB17

> DDL is not a mandatory stage before each SELECT, so authentication→DDL→DML permutations do not describe the processing sequence.

Study-note section ADB-S01

> A query tree corresponds to a relational-algebra expression: base relations appear at leaves and operators at internal nodes.

Related reading (context, not primary note proof)

Study-note section ADB23

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 29 | lines 9-20

> 1.6 Query Processing Steps
> • Processing can be divided into 4 :Decomposition, Optimization , Execution and Code generation
> 1.7 Query Decomposition
> • It is the process of transforming a high level query into a relational algebra query, and to check that the query is syntactically and semantically correct. It Consists of parsing and validation

Typical stages in query decomposition are:
i. Analysis: lexical and syntactical analysis of the query (correctness) based on attributes, data type.. ,. Query tree will be built for the query containing leaf node for base relations, one or many non-leaf nodes for relations produced by relational algebra operations and root node for the result of the query. Sequence of operation is from the leaves to the

Fundamental of Database Systems module | PDF page 44 | lines 6-13

3. Query Execution Plans
◦ An execution plan for a relational algebra query consists of :
◦ Relational algebra query tree
◦ Information about the access methods to be used for each relation
◦ The methods to be used in computing the relational operators stored in the tree.
1.9 Pipelining
● Pipelined evaluation: evaluate several operations simultaneously, passing the results of one operation on to the next.

Question sources

Computer Science MCQ practice collection · Question 237

Question 121 | AM0238

What is a cost model in query optimization?

A. A statistical model of query execution times

B. A model for estimating the cost of query plans

C. A model for analyzing database schemas

D. A statistical model for data distribution

Hint

Identify the resource objective, counts and unit costs before comparing execution strategies.

Answer B: A model for estimating the cost of query plans

The cost function combines I/O, CPU, communication and storage (p.41); optimization estimates and compares alternative strategy costs (p.42). A cost model therefore estimates costs of query plans.

Why each choice is right or wrong

A - Incorrect: Execution time can contribute, but this is narrower than the listed resource-cost model.

B - Correct: Estimating plan costs is the defining function.

C - Incorrect: Schema analysis is not evaluation of competing execution strategies.

D - Incorrect: Data distribution statistics are possible inputs, not the whole cost model.

Primary note proof

Study-note section ADB19

A cost model estimates resources for a candidate plan: page I/O, CPU, memory, temporary storage and network traffic. Catalog statistics include row/page counts, distinct values and distributions. Selection with fraction s predicts s×N rows. For non-NULL equijoin keys, a common estimate is |R|×|S|/max(V(R,k),V(S,k)), assuming uniform values and containment of the smaller distinct-value domain within the larger. Adjust counts for NULLs; if either non-NULL input is empty, the result is zero. Skew, correlations or incomplete domain overlap can invalidate this estimate; known disjoint domains give zero matches.

Study-note section ADB19

State the assumed cost objective before ranking strategies.

Study-note section ADB19

Disk I/O often dominates textbook local-disk examples; CPU can dominate in-memory work and network can dominate distributed plans.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 41 | lines 25-29

B. Cost Estimation Approach to Query Optimization
- The main idea is to minimize he cost of processing a query. The cost function is comprised of:
- I/O cost + CPU processing cost + communication cost + Storage cost
- These components might have different weights in different processing environments

Fundamental of Database Systems module | PDF page 42 | lines 10-25

Cost Estimation for RA Operations
- Many different ways of implementing RA operations.
- Aim of QO is to choose most efficient one.
- Use formulae that estimate costs for a number of options, and select one with lowest cost.
- Consider only cost of disk access, which is usually dominant cost in QP.
- Many estimates are based on cardinality of the relation, so need to be able to estimate this.

Using Selectivity and Cost Estimates in Query Optimization
- Cost-based query optimization:
  - Estimate and compare the costs of executing a query using different execution strategies and choose the strategy with the lowest cost estimate. (Compare to heuristic query optimization)
- Issues
  - Cost function
  - Number of execution strategies to be considered
- Cost Components for Query Execution

Question sources

Computer Science MCQ practice collection · Question 238

Question 122 | AM0239

What is a join operation in query processing?

A. Combining tuples from multiple tables based on a common attribute

B. Sorting the query result in ascending order

C. Grouping the query result based on a specific attribute

D. Filtering the query result based on a condition

Hint

Separate the query's representation from the operations used to calculate its result.

Answer A: Combining tuples from multiple tables based on a common attribute

The query joins Staff and Branch by matching branchNo (p.27). The heuristic combines a Cartesian product and selection on the join condition into a join (p.41); the distributed example similarly matches Employee.Dno and Department.Dnumber (p.120). A expresses this equijoin case; joins in general need not always use equality or identical attribute names.

Why each choice is right or wrong

A - Correct: Combining tuples that satisfy the matching attribute condition describes these joins.

B - Incorrect: Sorting changes order rather than combining source tuples.

C - Incorrect: Grouping builds groups/aggregates, not cross-relation matching.

D - Incorrect: Filtering alone selects rows rather than combining related tuples from tables.

Primary note proof

Study-note section ADB-S01

> A relation is represented by a table of tuples. Selection filters its rows by a predicate; projection retains chosen columns. A Cartesian product constructs every pair of input tuples. A matching join combines tuples satisfying its predicate, and can be represented as a selection over a product. Sorting orders results, while grouping collects rows sharing grouping values; these are different operations.

Study-note section ADB18

> Heuristics reduce intermediate work: push selective row filters toward base tables; replace product followed by a join predicate with a join; project unneeded columns away while retaining join/filter columns; reorder inner joins when equivalence permits; reuse common expressions when worthwhile.

Study-note section ADB18

> Cost-based optimization estimates candidate plans instead of relying on rules alone.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 27 | lines 4-15

What is Query Processing?
Steps required to transform high level SQL query into a correct and —efficient‖ strategy for execution and retrieval.
What is Query Optimization?
✓ The activity of choosing a single —efficient‖ execution strategy (from hundreds) as determined by database catalog statistics.
✓ The aim is to answer which relational algebra expression, equivalent to the given query, will lead to the most efficient solution plan?
✓ For each algebraic operator, what algorithm (of several available) do we use to compute that operator?
✓ How do operations pass data (main memory buffer, disk buffer,...)?
✓ Will this plan minimize resource usage? (CPU/Response Time/Disk)

Fundamental of Database Systems module | PDF page 41 | lines 25-29

B. Cost Estimation Approach to Query Optimization
- The main idea is to minimize he cost of processing a query. The cost function is comprised of:
- I/O cost + CPU processing cost + communication cost + Storage cost
- These components might have different weights in different processing environments

Fundamental of Database Systems module | PDF page 120 | lines 16-33

6.4 Query Processing in Distributed Databases
• Issues
- Cost of transferring data (files and results) over the network.
• This cost is usually high, so some optimization is necessary.
• Example: suppose there are three sites. Where the relation Employee at site 1, Department at Site 2 and no relation at site 3
- Employee at site 1. 10,000 rows. Row size = 100 bytes. Table size = 106
bytes.
- Department at Site 2. 100 rows. Row size = 35 bytes. Table size = 3,500 bytes.
- And a query is initiated from S3 to retrieve employees [First Name (15 byte long), Last name (15 byte long) and Department name (10 byte long) total of 40 bytes]
• Q: For each employee, retrieve employee Fname, Lname, and department name
• Q: δFname,Lname,Dname (Employee Dno = Dnumber Department)
• Assumption
Site 3

Fundamental of Database Systems module | PDF page 41 | lines 5-7

- Combine Cartesian product with subsequent Selection whose predicate represents join condition into a Join operation.
- Use associativity of binary operations to rearrange leaf nodes so leaf nodes with most

Question sources

Computer Science MCQ practice collection · Question 239

Question 123 | AM0240

Which listed index or clustered-storage approaches can speed a suitable query workload, without a claim that any always wins? Select all that apply.

A. B-tree index

B. Hash index

C. Bitmap index

D. Clustered index

Hint

Match the predicate and workload to the access structure or reasons to choose a plan again.

Answer A, B, C, D: A. B-tree index; B. Hash index; C. Bitmap index; D. Clustered index

B-tree indexes support ordered and range access, hash indexes target equality, and bitmap indexes can support analytic selections. Clustered storage places related rows close together and can reduce access work for suitable queries. Each approach can therefore help an appropriate workload. Space, maintenance and selectivity still matter, so accepting all four does not claim that every query benefits or that one design is universally best.

Why each choice is right or wrong

A - Correct: B-tree indexes support selective ordered and range access for suitable queries.

B - Correct: Hash indexes can accelerate equality lookups on the indexed key.

C - Correct: Bitmap indexes can accelerate suitable analytic selection workloads.

D - Correct: Clustered storage keeps related rows close and can reduce reads for appropriate access patterns.

Primary note proof

Study-note section ADB23

B/B+ tree indexes support ordered and range predicates; hash indexes target equality; bitmap indexes can help some analytic selections. Clustered storage keeps related rows close. An index may reduce reads but adds space and update work; scanning can be faster when most rows qualify. Commonly useful is not a unique property of one index type.

Study-note section ADB23

Avoid universal rules that joins always beat subqueries, functions can never use indexes or more indexes always improve performance.

Study-note section ADB23

Updating statistics does not itself execute the query.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 21 | lines 5-11

5.5 Index
Indexes are additional auxiliary access structures which are used to speed up the retrieval of records in response to certain search conditions. Index structures are additional files, which provide alternative ways to access the records without affecting the physical placement of records on disk.
Types of Indexes
Dense Index

Fundamental of Database Systems module | PDF page 22 | lines 11-20

Clustering Indexes
Here files are assumed as ordered with a Non-key field. If file records are physically ordered on a non-key field which doesn‘t have a distinct value for each record that field is called clustering field and the data file is called a clustered file.
We can create a different type of index, called a clustering index, to speedup retrieval of all records that have the same value for the clustering field. If the ordering field is not a key field i.e; if numerous records in the file can have the same value for the ordering field clustering index can be used.
A clustering index is also an ordered file with two fields. The first field is of the same type as the clustering field of the data file, and the second field is a disk block/cluster pointer.

Question sources

Computer Science MCQ practice collection · Question 240

Question 124 | AM0241

What is query re-optimization in a database system?

A. Re-executing the query with different parameters

B. Re-evaluating the query cost based on updated statistics

C. Re-writing the query in a different query language

D. Re-organizing the database schema to improve performance

Hint

Match the predicate and workload to the access structure or reasons to choose a plan again.

Answer B: Reconsider the plan using updated cost information or statistics.

ADB23 defines re-optimization as choosing a plan again when statistics, parameters or observed cardinalities suggest that the existing plan is unsuitable. B is the listed description of this cost reconsideration. Merely executing again with other values does not necessarily re-optimize, and changing the query language or redesigning the schema is not the definition.

Why each choice is right or wrong

A - Incorrect: Re-execution with new parameters can reuse the same plan and is not inherently re-optimization.

B - Correct: Updated statistics can change cost estimates and lead to choosing a new execution plan.

C - Incorrect: Changing query language is not required to reconsider an execution strategy.

D - Incorrect: Schema reorganization is a design change rather than the definition of query re-optimization.

Primary note proof

Study-note section ADB23

> B/B+ tree indexes support ordered and range predicates; hash indexes target equality; bitmap indexes can help some analytic selections. Clustered storage keeps related rows close. An index may reduce reads but adds space and update work; scanning can be faster when most rows qualify. Commonly useful is not a unique property of one index type.

Study-note section ADB23

> Re-optimization chooses a plan again when statistics, parameters or observed cardinalities suggest the old plan is unsuitable.

Study-note section ADB23

> Updating statistics does not itself execute the query.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 27 | lines 4-15

> What is Query Processing?
> Steps required to transform high level SQL query into a correct and —efficient‖ strategy for execution and retrieval.
> What is Query Optimization?
> ✓ The activity of choosing a single —efficient‖ execution strategy (from hundreds) as determined by database catalog statistics.
> ✓ The aim is to answer which relational algebra expression, equivalent to the given query, will lead to the most efficient solution plan?
> ✓ For each algebraic operator, what algorithm (of several available) do we use to compute that operator?

✓ How do operations pass data (main memory buffer, disk buffer,...)?
✓ Will this plan minimize resource usage? (CPU/Response Time/Disk)

Fundamental of Database Systems module | PDF page 42 | lines 10-25

Cost Estimation for RA Operations
- Many different ways of implementing RA operations.
- Aim of QO is to choose most efficient one.
- Use formulae that estimate costs for a number of options, and select one with lowest cost.
- Consider only cost of disk access, which is usually dominant cost in QP.
- Many estimates are based on cardinality of the relation, so need to be able to estimate this.

Using Selectivity and Cost Estimates in Query Optimization
- Cost-based query optimization:
  - Estimate and compare the costs of executing a query using different execution strategies and choose the strategy with the lowest cost estimate. (Compare to heuristic query optimization)
- Issues
  - Cost function
  - Number of execution strategies to be considered
- Cost Components for Query Execution

Question sources

Computer Science MCQ practice collection · Question 241

Question 125 | AM0242

What is query pipelining in query processing?

A. Dividing a query into subqueries and executing them in parallel

B. Dividing a query into stages and executing them sequentially

C. Dividing a query into multiple queries and executing them in parallel

D. Dividing a query into multiple stages and executing them concurrently

Hint

Distinguish overlapping dependent operators, independent workers and reused cached objects.

Answer D: Dividing a query into multiple stages and executing them concurrently

Pipelined evaluation evaluates several operations simultaneously and passes results directly from one to the next (p.44). D means overlapping producer/consumer stages, not necessarily executing independent whole queries.

Why each choice is right or wrong

A - Incorrect: Parallel independent subqueries are not the defining inter-operator tuple stream.

B - Incorrect: Pure sequential completion of each whole stage loses the stated simultaneous overlap.

C - Incorrect: Multiple independent queries are not the defining pipeline within an expression tree.

D - Correct: Concurrent overlapping stages passing tuples match the explicit definition.

Primary note proof

Study-note section ADB22

> Materialization stores an intermediate result before the next operator consumes it. Pipelining passes tuples as they become available, overlapping producer and consumer work and reducing temporary I/O. A sort is normally blocking; a hash join must build its hash table before probing can output, though later probe/output can stream. Pipelining does not require separate CPUs.

Study-note section ADB22

> Parallel execution uses multiple workers, for example partitioned scans/joins, independent subplans or operator pipelines.

Study-note section ADB22

> Specify the kind of query caching being discussed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 44 | lines 6-13

> 3. Query Execution Plans
> ◦ An execution plan for a relational algebra query consists of :
> ◦ Relational algebra query tree
> ◦ Information about the access methods to be used for each relation
> ◦ The methods to be used in computing the relational operators stored in the tree.
> 1.9 Pipelining
> • Pipelined evaluation: evaluate several operations simultaneously, passing the results of one operation on to the next.

Fundamental of Database Systems module | PDF page 44 | lines 12-20

> • Pipelined evaluation: evaluate several operations simultaneously, passing the results of one operation on to the next.
> • E.g., in expression tree, don‘t store result of
> ◦ Instead, pass tuples directly to the join.. Similarly, don‘t store result of join, pass tuples directly to projection.
> • Much cheaper than materialization: no need to store a temporary relation to disk.
> • Pipelining may not always be possible - e.g., sort, hash-join.
> • For pipelining to be effective, use evaluation algorithms that generate output tuples even as tuples are received for inputs to the operation.

Question sources

Computer Science MCQ practice collection · Question 242

Question 126 | AM0243

What is query result caching, specifically caching returned answers rather than execution plans or intermediate rows?

A. Storing query results for future reuse

B. Storing intermediate results during query execution

C. Storing database metadata for query optimization

D. Storing query plans for efficient execution

Hint

Distinguish overlapping dependent operators, independent workers and reused cached objects.

Answer A: A. Storing query results for future reuse

Result caching stores a query's returned rows or answer for future reuse. It needs suitable invalidation when underlying data changes. Plan caching instead stores the execution strategy without its returned rows. Intermediate rows retained during one execution are materialization, while database metadata supports optimization. Naming the cached object makes result caching distinct from these other useful storage mechanisms.

Why each choice is right or wrong

A - Correct: This stores final query answers for future reuse.

B - Incorrect: Intermediate-row storage is materialization rather than final-answer caching.

C - Incorrect: Metadata supports optimization but is not the query's returned answer.

D - Incorrect: Plan caching stores strategy rather than returned rows.

Primary note proof

Study-note section ADB22

> Parallel execution uses multiple workers, for example partitioned scans/joins, independent subplans or operator pipelines. Interleaving on one CPU is concurrency; simultaneous cores provide parallelism. Result caching reuses answers and needs invalidation when underlying data changes. Plan caching reuses an execution strategy, not the rows returned. Buffer caching keeps data pages; materialization retains intermediate rows. Specify the kind of query caching being discussed.

Study-note section ADB22

> Materialization stores an intermediate result before the next operator consumes it.

Study-note section ADB19

> Catalog statistics include row/page counts, distinct values and distributions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 41 | lines 25-29

B. Cost Estimation Approach to Query Optimization
- The main idea is to minimize he cost of processing a query. The cost function is comprised of:
- I/O cost + CPU processing cost + communication cost + Storage cost
- These components might have different weights in different processing environments

Fundamental of Database Systems module | PDF page 104 | lines 19-25

• For Undo: Before a data item‘s AFIM is flushed to the database disk (overwriting the BFIM) its BFIM must be written to the log and the log must be saved on a stable store (log disk).
• For Redo: Before a transaction executes its commit operation, all its AFIMs must be written to the log and the log must be saved on a stable store.
• Standard Recovery Terminology

Fundamental of Database Systems module | PDF page 41 | lines 11-15

- Compute common expressions once.
- If common expression appears more than once, and result not too large, store result and reuse it when required.
- Useful when querying views, as same expression is used to construct view each time.

Summary of Heuristics for Algebraic Optimization:

Fundamental of Database Systems module | PDF page 104 | lines 2-13

Data Caching
• Data items to be modified are first stored into database cache by the Cache Manager (CM) and after modification they are flushed (written) to the disk
• When DBMS request for read /write operation on some item
• It check the requested data item is in the cache or not
• If it is not, the appropriate disk block are copied to the cache
• If the cache is already full, some buffer replacement policy can be used .
Like
• Least recent used (LRU)
• FIFO
• While replacing buffers , first of all the updated value on that buffer should be saved on the appropriate block in the data base

Question sources

Computer Science MCQ practice collection · Question 243

Question 127 | AM0244

What is query parallelization in query processing?

A. Dividing a query into subqueries and executing them in parallel

B. Dividing a query into stages and executing them sequentially

C. Dividing a query into multiple queries and executing them in parallel

D. Dividing a query into multiple stages and executing them concurrently

Hint

Distinguish overlapping dependent operators, independent workers and reused cached objects.

Answer A: Dividing a query into subqueries and executing them in parallel

The fragmentation discussion says a transaction can be divided into several subqueries operating on fragments, increasing concurrency (p.117). A matches this parallel decomposition; it is distinct from pipelining a dependent stream of stages.

Why each choice is right or wrong

A - Correct: Independent fragment subqueries can run in parallel as described.

B - Incorrect: Sequential stages do not supply parallelism.

C - Incorrect: Replacing one query with multiple separate whole queries is not the stated subquery decomposition.

D - Incorrect: Concurrent stages describe pipelining when results flow between dependent operators, not the specific fragment-subquery approach.

Primary note proof

Study-note section ADB22

> Materialization stores an intermediate result before the next operator consumes it. Pipelining passes tuples as they become available, overlapping producer and consumer work and reducing temporary I/O. A sort is normally blocking; a hash join must build its hash table before probing can output, though later probe/output can stream. Pipelining does not require separate CPUs.

Study-note section ADB17

> Query processing is the whole activity; selecting a suitable efficient strategy is optimization.

Study-note section ADB22

> Parallel execution uses multiple workers, for example partitioned scans/joins, independent subplans or operator pipelines.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 117 | lines 15-31

6.2 Data Replication and Fragmentation: Distributed data storage
• There are two approaches to store the relation in the distributed database : Replication and Fragmentation
I. Data Replication
• The system maintain several identical copies of the relation & store each copy at a different site
• In general it enhance the performance of read operation and increase the availability of data to read only transaction. However, update transactions incur greater overhead
II. Data Fragmentation
- Split a relation into logically related and correct parts.
- The main reasons for fragmenting a relation are
• Efficiency- data that is not needed by the local applications is not stored
• Parallelism - a transaction can be divided into several subqueries that operate on fragments which will increase the degree of concurrency
- but reconstruction of the whole relation will require accessing data from all sites containing part of the relation

Question sources

Computer Science MCQ practice collection · Question 244

Question 128 | CS0066

If power went off while processing a transaction, what should the DBMS do to reverse any changes made to the database?

A. Read_transaction

B. Fail transaction

C. Commit transaction

D. Rollback

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer D: Rollback

To reverse changes of an incomplete failed transaction, recovery undoes its effects (p.103) by restoring before images (p.107). This is rollback. A committed transaction instead requires durability, so the answer assumes the interrupted transaction had not committed, as implied by reverse any changes.

Why each choice is right or wrong

A - Incorrect: Reading a transaction is not an undo operation.

B - Incorrect: Marking failure does not itself restore old values.

C - Incorrect: Committing would preserve rather than reverse the changes.

D - Correct: Rollback/undo is the direct reversal mechanism.

Primary note proof

Study-note section ADB27

> Write-ahead logging requires the update log needed for recovery to reach stable storage before the corresponding changed data page reaches disk. To acknowledge a durable commit, force the transaction's required log records, including its commit record, to stable storage. A buffer can be changed before that log flush; WAL orders durable writes, not every CPU assignment.

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward.

Study-note section ADB06

> Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Fundamental of Database Systems module | PDF page 97 | lines 4-13

> i. Purpose of Database Recovery
> ◦ To bring the database into the last consistent state, which existed prior to the failure.
> ◦ To preserve transaction properties (Atomicity &and Durability).
> • The recovery manager of a DBMS is responsible to ensure
> • atomicity by undoing the action of transaction that do not commit
> • Durability by making sure that all actions of committed transaction survive system crash
> ii. Types of Failure
> ◦ The database may become unavailable for use due to :

Fundamental of Database Systems module | PDF page 103 | lines 6-23

> • Recovery from catastrophic
> • If there is extensive damage to the wide portion of the database
> • This method restore a past copy of the database from the backup storage and reconstructs operation of a committed transaction from the back up log up to the time of failure
> • Recovery from non-catastrophic failure
> • When the database is not physically damaged but has be come in consistent
> • The strategy uses undoing and redoing some operations in order to restore to a consistent state : example
> • For instance,

• If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to redo (roll forward) transaction‘s updates.
• If transaction had not committed at failure time, recovery manager has to undo (rollback) any effects of that transaction for atomicity. Example,
Transaction Log
◦ For recovery from any type of failure data values prior to modification (BFIM -

Fundamental of Database Systems module | PDF page 107 | lines 8-23

Transaction Roll-back (Undo) and Roll-Forward (Redo)
• If transaction fails for what so ever reason after updating the data base it must be necessary to roll back
• If a transaction T is roll back , any transaction that has read the value of some data item X written by T must be also be roll back
• Similarly once S is roll back, any transaction R that has read the values of some data item Y written by S must aslo be roll back, and so on..
• To maintain such cases, a transaction‘s operations are redone or undone.
• Undo: Restore all BFIMs on to disk (Remove all AFIMs).
• Redo: Restore all AFIMs on to disk.
• Database recovery is achieved either by performing only Undos or only Redos or by a combination of the two. These operations are recorded in the log as they happen.
Data Update and Recovery Scheme : Three types
i. Deferred Update (No Undo/Redo)
◦ The data update goes as follows:

Question sources

Computer Science Exit Exam practice collection · Question 66

Question 129 | NS0051

What is the main difference between homogeneous and heterogeneous distributed databases?

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer: Homogeneous systems use compatible DBMS technology; heterogeneous systems integrate differing DBMSs, models, schemas or interfaces.

ADB33 defines the distinction through compatibility of database technology and representation. Operating-system differences can occur but do not define heterogeneity. Homogeneous sites need not have identical schemas or automatic replication. Heterogeneous integration requires explicit mappings for names, types and semantics, rather than assuming that common connectivity alone reconciles their meanings.

Primary note proof

Study-note section ADB33

> A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB33

> ODBC and JDBC provide connectivity interfaces; they do not automatically solve semantic differences or create one distributed transaction.

Study-note section ADB33

> Heterogeneous design must reconcile naming, types, units, identifiers, query capabilities and transaction semantics.

Question sources

Advanced Database Systems review questions · Question 1

Question 130 | NS0052

Why is data fragmentation useful in distributed databases?

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer: Fragmentation improves locality, manageable placement and parallelism by splitting data across useful site boundaries.

ADB36 splits rows with predicates and columns with projections while preserving reconstruction through unions or key joins. This can place frequently used data near its consumers and allow independent work on pieces. Correct fragments must preserve completeness and required disjointness. Fragmentation alone creates no spare copy, so losing the only site containing a fragment still makes it unavailable.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB36

> Partitioning improves locality and parallelism but alone creates no spare copy.

Study-note section ADB36

> Losing the only site containing a fragment makes that fragment unavailable, even while other sites continue.

Question sources

Advanced Database Systems review questions · Question 2

Question 131 | NS0053

What is the purpose of two-phase locking (2PL) in concurrency control?

Hint

Track the first lock release and the transaction-completion boundary.

Answer: 2PL guarantees conflict serializability by separating lock acquisition from release.

During the growing phase a transaction acquires or upgrades locks; after it releases any lock, the shrinking phase forbids acquiring new ones. For well-formed transactions this guarantees conflict serializability. Basic 2PL can still deadlock and permit cascading aborts, so strict or conservative variants add different protections. It is distinct from distributed two-phase commit.

Primary note proof

Study-note section ADB12

> Basic 2PL has a growing phase that acquires/upgrades locks and a shrinking phase that releases/downgrades them. Once any lock is released, no new lock may be acquired. The lock point is the instant the final lock is obtained. 2PL guarantees conflict serializability for well-formed transactions, but can deadlock and basic 2PL alone need not prevent cascading aborts.

Study-note section ADB12

> Two-phase locking orders access to data; two-phase commit coordinates a distributed commit decision.

Study-note section ADB12

> Strict 2PL retains exclusive locks until commit/abort.

Question sources

Advanced Database Systems review questions · Question 3

Question 132 | NS0054

Why is write-ahead logging important for database recovery?

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer: WAL makes recovery information durable before its changed data pages, enabling safe undo/redo after crashes.

ADB27 requires needed update logs to reach stable storage before corresponding changed pages. A durable commit also forces required log records, including commit. Recovery can then use before-images for undo and after-images for redo if buffered data was lost. WAL orders durable writes; modifying an in-memory buffer before the log flush does not itself violate it.

Primary note proof

Study-note section ADB27

> Write-ahead logging requires the update log needed for recovery to reach stable storage before the corresponding changed data page reaches disk. To acknowledge a durable commit, force the transaction's required log records, including its commit record, to stable storage. A buffer can be changed before that log flush; WAL orders durable writes, not every CPU assignment.

Study-note section ADB27

> Redo installs logged after-images when necessary, usually forward.

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward.

Question sources

Advanced Database Systems review questions · Question 4

Question 133 | NS0055

How does a data warehouse differ from an operational database?

Hint

Distinguish operational transactions, analytical workloads and physical storage layout.

Answer: A warehouse integrates historical data for analysis; an operational database serves frequent business transactions.

OLTP handles account updates and other frequent operational transactions, while OLAP scans and summarizes large datasets. A warehouse commonly integrates sources through extraction, transformation and loading for historical analysis. These are workload tendencies rather than absolute restrictions: operational databases may retain history, and warehouses may receive updates. Schema and storage choices should follow the actual workload.

Primary note proof

Study-note section ADB41

> OLTP handles frequent operational transactions, such as account updates; OLAP scans and summarizes large datasets for analysis. A warehouse integrates data for historical analysis, often through extraction, transformation and loading. These are workload tendencies: OLTP can retain history and warehouses can be updated.

Question sources

Advanced Database Systems review questions · Question 5

Question 134 | NS0056

What does ACID stand for in the context of database transactions?

A. Atomicity, Consistency, Isolation, Durability

B. Accuracy, Consistency, Integrity, Durability

C. Atomicity, Concurrency, Isolation, Data

D. Availability, Consistency, Isolation, Durability

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer A: Atomicity, Consistency, Isolation, Durability.

ADB06 names all four ACID properties and assigns distinct roles: all-or-none effects, preserved invariants, controlled interference and durable commits. Only A lists those names in the correct order. Accuracy, integrity, concurrency, data and availability are related database concepts but do not replace the letters of this established transaction acronym.

Why each choice is right or wrong

A - Correct: It lists the four ACID properties in the proper acronym order.

B - Incorrect: Accuracy and Integrity replace Atomicity and Isolation, changing the defined acronym.

C - Incorrect: Concurrency and Data incorrectly replace Consistency and Durability.

D - Incorrect: Availability is not the A in the ACID transaction acronym.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Question sources

Advanced Database Systems review questions · Question 1

Question 135 | NS0057

Which ACID property ensures that a transaction is treated as an indivisible unit - either all steps succeed or none are applied?

A. Consistency

B. Isolation

C. Atomicity

D. Durability

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer C: Atomicity.

The question specifies indivisibility of the transaction effects: partial success must not remain if the transaction fails. ADB06 assigns that guarantee to atomicity. Consistency concerns valid invariants, isolation controls interference, and durability protects already committed work. Those properties support correctness too, but none directly means all-or-none effects.

Why each choice is right or wrong

A - Incorrect: Consistency concerns preserving integrity rules rather than specifically making effects indivisible.

B - Incorrect: Isolation controls interactions between concurrent transactions rather than partial transaction application.

C - Correct: Atomicity applies every transaction effect together or discards the incomplete work.

D - Incorrect: Durability protects committed effects after failure rather than defining all-or-none completion.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Question sources

Advanced Database Systems review questions · Question 2

Question 136 | NS0058

Account A = 5,000 Birr, Account B = 8,000 Birr. After transferring 750 Birr from A to B, which values maintain CONSISTENCY?

A. A = 4,000, B = 9,000

B. A = 4,250, B = 8,750

C. A = 5,750, B = 7,250

D. A = 4,250, B = 8,000

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: A = 4,250 Birr and B = 8,750 Birr.

The requested transfer subtracts 750 from 5000 and adds 750 to 8000, giving 4250 and 8750. The invariant total remains 13000. A and C also sum to 13000, but implement a different amount or the reverse direction. Preserving the total alone does not implement the specified successful transfer; D loses the credit.

Why each choice is right or wrong

A - Incorrect: The sum remains 13000, but the changes transfer 1000 rather than the requested 750.

B - Correct: It debits 750 from A, credits 750 to B and preserves the 13000 total.

C - Incorrect: It preserves the total but transfers 750 in the reverse direction.

D - Incorrect: It debits A without crediting B, reducing the total to 12250.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Study-note section ADB06

> Unchanged or reverse-transfer balances can also preserve that sum; only the requested debit/credit implements the specified successful transfer.

Study-note section ADB06

> A=5,000 and B=8,000 become 4,250 and 8,750 after transferring 750 from A to B.

Question sources

Advanced Database Systems review questions · Question 3

Question 137 | NS0059

Which ACID property guarantees that committed transaction changes persist even after a system crash?

A. Atomicity

B. Consistency

C. Isolation

D. Durability

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer D: Durability.

ADB06 defines durability as preserving committed effects despite failure. A system may leave data pages buffered at commit provided durable recovery information can restore the changes after a crash. Atomicity controls partial outcomes, consistency protects declared invariants and isolation controls concurrent interference; they do not name this persistence guarantee.

Why each choice is right or wrong

A - Incorrect: Atomicity means all-or-none effects, not specifically persistence after a successful commit.

B - Incorrect: Consistency concerns valid integrity rules before and after correct work.

C - Incorrect: Isolation controls concurrent interference rather than persistence across a crash.

D - Correct: Durability ensures committed changes survive failures through the supported recovery mechanisms.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Question sources

Advanced Database Systems review questions · Question 4

Question 138 | NS0060

In the Read_item(X) operation, what is the THIRD and final step?

A. Store the updated block back to disk

B. Find the disk block address containing X

C. Copy the disk block into main memory buffer

D. Copy item X from the buffer to the program variable named x

Hint

Trace movement between program variables, buffers and disk, or the meaning of a recorded transaction event.

Answer D: Copy X from the buffer into the program variable x.

ADB07 first locates the page containing X, brings it into a buffer if absent and then copies X into a program variable. Thus D is the final conceptual read step. Locating and loading precede it, and storing an updated block to disk belongs to writing/flushing rather than completing read_item. A buffer hit can avoid physical I/O.

Why each choice is right or wrong

A - Incorrect: A read does not update and write the block back to disk.

B - Incorrect: Locating the containing page is the first conceptual step, not the final one.

C - Incorrect: Loading the page into a buffer precedes copying the item into the program variable.

D - Correct: The final read step copies the requested item from its buffer into x.

Primary note proof

Study-note section ADB07

> read_item(X) locates X's page, brings it into a buffer if absent, and copies X into a program variable. write_item(X) copies the new program value into the buffered page, marking it dirty. Writing that page to disk can occur later according to the buffer policy; write_item is not inherently an immediate disk flush. Insert and delete also modify data; sorting is a query-processing operation rather than a primitive item update.

Question sources

Advanced Database Systems review questions · Question 5

Question 139 | NS0061

A transaction that has started and is currently executing its operations is in which state?

A. Committed

B. Partially Committed

C. Active

D. Failed

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer C: Active.

ADB06 defines Active as the state while transaction operations are executing. Partially committed follows completion of the last operation but precedes assured durable commitment. Committed means commitment succeeded, while Failed means execution encountered a failure. Merely starting and continuing its operations therefore places this transaction in Active.

Why each choice is right or wrong

A - Incorrect: Committed requires successful commitment, which has not occurred while operations are still executing.

B - Incorrect: Partially committed follows the last operation rather than ordinary ongoing execution.

C - Correct: Active is the state during execution of the transaction operations.

D - Incorrect: Failed requires an execution failure, which the stem does not report.

Primary note proof

Study-note section ADB06

> Active means operations are executing. Partially committed means the last operation finished but durable commitment is not yet assured. Committed means commitment succeeded. A failure leads to Failed, then Aborted after rollback; a transaction may terminate or be restarted as a new attempt. Concurrency control primarily enforces isolation; recovery supports atomicity and durability. Transaction management is the broader subsystem, so both it and recovery management can be legitimate answers to broad responsibility questions.

Question sources

Advanced Database Systems review questions · Question 6

Question 140 | NS0062

Which transaction state means all operations executed successfully but the commit has NOT been finalized yet?

A. Active

B. Partially Committed

C. Committed

D. Aborted

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: Partially committed.

Finishing all operations does not yet establish durable commitment. ADB06 names this intermediate state Partially committed. The transaction becomes Committed only after commitment succeeds, or can fail before that boundary. Active describes operations still running, while Aborted means rollback completed, so neither matches successful operations awaiting finalized commit.

Why each choice is right or wrong

A - Incorrect: Active means operations are still executing, whereas the last operation has finished here.

B - Correct: Partially committed means execution finished but durable commitment is not yet assured.

C - Incorrect: Committed requires the commit itself to have succeeded, which the stem excludes.

D - Incorrect: Aborted follows rollback rather than successful execution awaiting a commit decision.

Primary note proof

Study-note section ADB06

> Active means operations are executing. Partially committed means the last operation finished but durable commitment is not yet assured. Committed means commitment succeeded. A failure leads to Failed, then Aborted after rollback; a transaction may terminate or be restarted as a new attempt. Concurrency control primarily enforces isolation; recovery supports atomicity and durability. Transaction management is the broader subsystem, so both it and recovery management can be legitimate answers to broad responsibility questions.

Question sources

Advanced Database Systems review questions · Question 7

Question 141 | NS0063

What does the log record [abort, T] indicate?

A. Transaction T has just started execution

B. Transaction T committed successfully

C. Transaction T was aborted and ALL its changes must be discarded

D. Transaction T read a data item

Hint

Trace movement between program variables, buffers and disk, or the meaning of a recorded transaction event.

Answer C: T is aborting/aborted and its transaction effects must be discarded.

ADB07 includes ABORT among conceptual log records and warns that it can mark the start of rollback rather than prove every undo step is already complete. C is correct because it says the changes must be discarded, not that all undo has finished. START, COMMIT and ordinary read activity have different meanings.

Why each choice is right or wrong

A - Incorrect: A START record, rather than ABORT, marks beginning transaction execution.

B - Incorrect: COMMIT records successful completion; ABORT means the opposite outcome.

C - Correct: Abort requires discarding transaction effects, though this record need not prove all undo is already complete.

D - Incorrect: The record denotes abort handling rather than a read_item operation.

Primary note proof

Study-note section ADB07

> A conceptual log can contain START, UPDATE(T,X,old,new), COMMIT, ABORT and END records. Before-images support undo; after-images support redo. Real systems may use physical or logical records, and need not log every read. COMMIT records successful completion; an ABORT record can mark the start of rollback rather than prove every undo step is complete.

Question sources

Advanced Database Systems review questions · Question 8

Question 142 | NS0064

Which of the following is NOT one of the four basic transaction operations on a database?

A. Read

B. Write

C. Sort

D. Delete

Hint

Trace movement between program variables, buffers and disk, or the meaning of a recorded transaction event.

Answer C: Sort.

ADB07 distinguishes read_item and write_item and notes that inserts and deletes also modify data. Sorting is instead an operation used in query processing, not a primitive item update in this classification. The exact count four is a teaching convention, but the note directly supplies the relevant distinction that singles out Sort.

Why each choice is right or wrong

A - Incorrect: Reading is an explicitly defined primitive database item operation.

B - Incorrect: Writing is an explicitly defined primitive database item modification operation.

C - Correct: Sorting is a query-processing operation rather than the primitive item access/update category.

D - Incorrect: Delete modifies stored data and belongs with the listed primitive data operations.

Primary note proof

Study-note section ADB07

> read_item(X) locates X's page, brings it into a buffer if absent, and copies X into a program variable. write_item(X) copies the new program value into the buffered page, marking it dirty. Writing that page to disk can occur later according to the buffer policy; write_item is not inherently an immediate disk flush. Insert and delete also modify data; sorting is a query-processing operation rather than a primitive item update.

Question sources

Advanced Database Systems review questions · Question 9

Question 143 | NS0065

The Isolation property of ACID ensures that concurrent transactions:

A. Always read the same data at the same time

B. Do not interfere with each other; each executes independently

C. Commit in the same order they started

D. Always complete successfully without any errors

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: Concurrent work is isolated from improper interference.

ADB06 defines isolation as controlling interference between concurrent transactions. B captures logical independence under the specified isolation guarantee; it does not mean transactions literally use separate hardware or never wait. Isolation does not require identical reads, start-order commits or universal successful completion. Strong serializable isolation makes behavior equivalent to some serial order.

Why each choice is right or wrong

A - Incorrect: Isolation does not require all transactions to read identical values at the same time.

B - Correct: Isolation controls interference so concurrent transactions retain the promised logical execution semantics.

C - Incorrect: Isolation does not require commit order to match transaction start order.

D - Incorrect: Conflicts and failures can still cause aborts; isolation is not a success guarantee.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Study-note section ADB09

> A serializable schedule is equivalent to some serial order; it may interleave transactions.

Study-note section ADB09

> A serial schedule executes each transaction's operations consecutively.

Question sources

Advanced Database Systems review questions · Question 10

Question 144 | NS0066

In a simplified teaching sequence, write_item(X) updates a buffered page and then the buffer manager immediately flushes that page, after required WAL records are durable. What is the final physical step of this specified sequence?

A. Copy item X to program variable

B. Find the disk block address

C. Store the updated buffer block back to disk

D. Read the old value of X first

Hint

Trace movement between program variables, buffers and disk, or the meaning of a recorded transaction event.

Answer C: C. Store the updated buffer block back to disk

After X's program value has been copied into the buffered page, the specified sequence completes by writing the changed page back to disk. That is C. In ordinary buffering this flush may be deferred, but the question explicitly includes it. Locating a page or reading its old value occurs before or separately from the final physical output, and copying into a program variable describes reading.

Why each choice is right or wrong

A - Incorrect: Copying into a program variable is a read operation.

B - Incorrect: The page address is located before the final output.

C - Correct: The explicitly included flush stores the updated buffered page on disk.

D - Incorrect: Reading the old value is not the final physical output step.

Primary note proof

Study-note section ADB07

> read_item(X) locates X's page, brings it into a buffer if absent, and copies X into a program variable. write_item(X) copies the new program value into the buffered page, marking it dirty. Writing that page to disk can occur later according to the buffer policy; write_item is not inherently an immediate disk flush. Insert and delete also modify data; sorting is a query-processing operation rather than a primitive item update.

Study-note section ADB27

> A buffer can be changed before that log flush; WAL orders durable writes, not every CPU assignment.

Study-note section ADB27

> To acknowledge a durable commit, force the transaction's required log records, including its commit record, to stable storage.

Question sources

Advanced Database Systems review questions · Question 11

Question 145 | NS0067

A transaction that encountered an error during execution and could not complete is in which state?

A. Aborted

B. Failed

C. Terminated

D. Partially Committed

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: Failed.

ADB06 places a transaction in Failed when an error prevents successful execution. After rollback removes its effects it reaches Aborted. The stem reports the execution error but does not say rollback has completed, so Failed is the precise state. Partially committed follows successful final operation execution, not an error that prevents completion.

Why each choice is right or wrong

A - Incorrect: Aborted requires completed rollback, which is not stated after the execution error.

B - Correct: Failed is the immediate state when an error prevents completing the transaction.

C - Incorrect: Termination is later lifecycle handling rather than the immediate error state.

D - Incorrect: Partially committed follows completion of the last operation rather than failed execution.

Primary note proof

Study-note section ADB06

> Active means operations are executing. Partially committed means the last operation finished but durable commitment is not yet assured. Committed means commitment succeeded. A failure leads to Failed, then Aborted after rollback; a transaction may terminate or be restarted as a new attempt. Concurrency control primarily enforces isolation; recovery supports atomicity and durability. Transaction management is the broader subsystem, so both it and recovery management can be legitimate answers to broad responsibility questions.

Question sources

Advanced Database Systems review questions · Question 12

Question 146 | NS0068

Transaction log records serve primarily for which purpose?

A. Speeding up query execution

B. Encrypting sensitive data

C. Enabling database recovery after failures by recording all changes

D. Enforcing access control permissions

Hint

Trace movement between program variables, buffers and disk, or the meaning of a recorded transaction event.

Answer C: Enable recovery by recording the information needed to undo or redo changes.

ADB07 records starts, updates, commits and aborts, with before-images for undo and after-images for redo. That makes recovery the primary purpose among the choices. Actual log formats may be physical or logical and need not record every read, so all changes should be understood as the change information required by the recovery design, not every operation indiscriminately.

Why each choice is right or wrong

A - Incorrect: The primary log purpose here is recovery rather than speeding query execution.

B - Incorrect: Transaction logging does not inherently encrypt the stored sensitive data.

C - Correct: Logged update and transaction-state information supports recovery of valid committed state after failures.

D - Incorrect: Access privileges control authorization; transaction log records are not themselves that enforcement mechanism.

Primary note proof

Study-note section ADB07

> A conceptual log can contain START, UPDATE(T,X,old,new), COMMIT, ABORT and END records. Before-images support undo; after-images support redo. Real systems may use physical or logical records, and need not log every read. COMMIT records successful completion; an ABORT record can mark the start of rollback rather than prove every undo step is complete.

Study-note section ADB24

> A user who can view but cannot change a record has restricted access privileges.

Study-note section ADB24

> Authentication establishes identity; authorization/access control determines permitted actions.

Question sources

Advanced Database Systems review questions · Question 13

Question 147 | NS0069

Which processing type uses multiple CPUs to execute different transactions simultaneously, achieving higher throughput?

A. Sequential processing

B. Interleaved processing

C. Parallel processing

D. Batch processing

Hint

Distinguish overlapping dependent operators, independent workers and reused cached objects.

Answer C: Parallel processing.

ADB22 distinguishes concurrency by interleaving on one CPU from actual simultaneous work on multiple cores. The stem explicitly uses multiple CPUs to execute different transactions at the same time, making Parallel the correct classification. Batch describes grouping work, and sequential processing performs work in order; neither guarantees the stated simultaneous execution.

Why each choice is right or wrong

A - Incorrect: Sequential processing does not execute different transactions simultaneously on multiple CPUs.

B - Incorrect: Interleaving alternates execution and need not provide simultaneous CPU work.

C - Correct: Multiple CPUs working simultaneously provide parallel execution.

D - Incorrect: Batch describes grouped submission or handling, not necessarily simultaneous multi-CPU execution.

Primary note proof

Study-note section ADB22

> Materialization stores an intermediate result before the next operator consumes it. Pipelining passes tuples as they become available, overlapping producer and consumer work and reducing temporary I/O. A sort is normally blocking; a hash join must build its hash table before probing can output, though later probe/output can stream. Pipelining does not require separate CPUs.

Study-note section ADB22

> Interleaving on one CPU is concurrency; simultaneous cores provide parallelism.

Study-note section ADB22

> Parallel execution uses multiple workers, for example partitioned scans/joins, independent subplans or operator pipelines.

Question sources

Advanced Database Systems review questions · Question 14

Question 148 | NS0070

Why are transactions important in database systems?

A. They make queries run faster

B. They ensure the database remains consistent and reliable through ACID properties

C. They allow the database to store more data

D. They simplify the SQL syntax for users

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: Transactions support consistent, reliable work through ACID guarantees.

ADB06 groups operations into a logical unit with all-or-none effects, invariant preservation, isolation and durable commits. These guarantees support consistent and reliable database behavior when application logic is correct. They do not inherently speed queries, expand storage capacity or simplify syntax. In particular a DBMS cannot make an incorrect business operation consistent merely by wrapping it in a transaction.

Why each choice is right or wrong

A - Incorrect: Transactions can add coordination costs and do not inherently make queries faster.

B - Correct: ACID guarantees support valid and recoverable logical work, provided business logic is correct.

C - Incorrect: Transaction semantics do not increase the database's physical storage capacity.

D - Incorrect: Grouping work atomically does not inherently simplify the SQL language for users.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Question sources

Advanced Database Systems review questions · Question 15

Question 149 | NS0071

T1 reads balance=$500 and adds $100, planning to write $600. T2 reads the same $500, adds $200, and writes $700. T1 then writes $600. What problem occurred?

A. Dirty Read

B. Lost Update (Write-Write Problem)

C. Incorrect Summary Problem

D. Deadlock

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer B: Lost update: T1 overwrites T2’s increment with its stale calculation.

Both transactions read 500. T2 writes 700 after adding 200, but T1 then writes its previously computed 600, erasing T2’s increment. A serial execution would produce 800. This matches the lost-update trace in ADB08; no waiting cycle or aggregate calculation is described, and reading an uncommitted value is not the stated cause.

Why each choice is right or wrong

A - Incorrect: Neither transaction is described as reading the other's uncommitted write.

B - Correct: T1's stale write of 600 overwrites T2's added 200; the correct combined result is 800.

C - Incorrect: No aggregate over mutually inconsistent versions is being calculated.

D - Incorrect: No circular waiting or blocked resource requests are present.

Primary note proof

Study-note section ADB08

> Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Study-note section ADB13

> Deadlock requires a waiting cycle: T1 holds X and wants Y; T2 holds Y and wants X.

Study-note section ADB08

> Incorrect summary mixes mutually inconsistent versions: a transfer moves 10 from A=100 to B=100; a sum reading old A=100 and new B=110 gets 210 instead of the invariant 200.

Question sources

Advanced Database Systems review questions · Question 16

Question 150 | NS0072

A dirty read occurs when transaction T1 reads data that:

A. T1 itself wrote earlier in the same transaction

B. T2 modified but has NOT yet committed, and T2 may later abort

C. Was permanently deleted from the database

D. Was last read more than one hour ago

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer B: Read another transaction’s uncommitted modification.

ADB08 defines dirty read as consuming a value written by an uncommitted other transaction. If that writer aborts, the dependent read used a value that should not survive. Reading one’s own changes is allowed and is not this anomaly; the age of a value or a completed permanent deletion does not define dirty reading.

Why each choice is right or wrong

A - Incorrect: Reading changes made earlier by the same transaction is not a dirty read.

B - Correct: The read depends on T2's uncommitted update, which may later need undo.

C - Incorrect: Permanent deletion is not the defining uncommitted-value dependency.

D - Incorrect: Elapsed time since a read does not determine whether the writer committed.

Primary note proof

Study-note section ADB08

> Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Study-note section ADB08

> A phantom means repeating a predicate query finds a changed qualifying set after another transaction inserts, deletes or changes matching rows.

Study-note section ADB08

> A single committed update from B=20 to B=50 does not alone prove an invalid summary of 90.

Question sources

Advanced Database Systems review questions · Question 17

Question 151 | NS0073

Which lock type allows MULTIPLE transactions to simultaneously READ a data item?

A. Exclusive Lock

B. Write Lock

C. Shared Lock

D. Binary Lock

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer C: Shared lock.

ADB11 states that a shared lock permits reading and multiple shared holders can coexist on the same item. Exclusive/write locks conflict with another transaction’s shared or exclusive lock on that item. Thus shared locking provides the requested concurrent-reader behavior. A generic binary locked/unlocked label does not establish shared-reader compatibility in the supplied model.

Why each choice is right or wrong

A - Incorrect: An exclusive lock conflicts with other transactions' read locks on the same item.

B - Incorrect: A write/exclusive lock does not grant multiple independent readers the requested shared ownership.

C - Correct: Shared locks are compatible with other shared locks, permitting multiple readers.

D - Incorrect: The binary label does not provide the explicit S/S compatibility defined in the note.

Primary note proof

Study-note section ADB11

> A shared (S) lock permits reading; multiple S locks on the same item can coexist. An exclusive (X) lock permits its holder to read and write and conflicts with another S or X on that item. Locks on different items do not conflict merely because one is exclusive. These are the basic lock-based rules; multiversion systems can permit additional snapshot-reader behavior.

Study-note section ADB-S04

> Shared/exclusive locking adds modes so multiple shared readers can coexist while incompatible exclusive access is excluded.

Study-note section ADB11

> One transaction may request a lock on an already locked item: it is granted if compatible, otherwise waits or aborts under the policy.

Question sources

Advanced Database Systems review questions · Question 18

Question 152 | NS0074

In Two-Phase Locking (2PL), a transaction can acquire new locks ONLY during which phase?

A. Shrinking Phase

B. Termination Phase

C. Growing Phase

D. Validation Phase

Hint

Track the first lock release and the transaction-completion boundary.

Answer C: Growing phase.

Basic 2PL separates acquiring/upgrading locks from releasing/downgrading them. ADB12 permits acquisition only in the growing phase; after any release, no new lock may be acquired. Termination is a transaction boundary and validation belongs to a different control approach. The growing/shrinking rule is what distinguishes 2PL from unrestricted use of locks.

Why each choice is right or wrong

A - Incorrect: The shrinking phase releases or downgrades locks and forbids new acquisitions.

B - Incorrect: Termination is not the 2PL phase in which new locks are acquired.

C - Correct: Growing permits acquiring and upgrading locks before any release begins.

D - Incorrect: Validation is an optimistic-control stage, not the lock-acquisition phase of 2PL.

Primary note proof

Study-note section ADB12

> Basic 2PL has a growing phase that acquires/upgrades locks and a shrinking phase that releases/downgrades them. Once any lock is released, no new lock may be acquired. The lock point is the instant the final lock is obtained. 2PL guarantees conflict serializability for well-formed transactions, but can deadlock and basic 2PL alone need not prevent cascading aborts.

Study-note section ADB12

> Two-phase locking orders access to data; two-phase commit coordinates a distributed commit decision.

Study-note section ADB16

> Optimistic/validation control reads and computes privately, validates its read/write sets against overlapping transactions, then publishes if validation succeeds; otherwise it retries.

Question sources

Advanced Database Systems review questions · Question 19

Question 153 | NS0075

What does Two-Phase Locking (2PL) guarantee that locking alone does NOT guarantee?

A. That no deadlocks will occur

B. Serializability - results equivalent to some serial execution

C. That all transactions complete without aborting

D. That every transaction reads the most current data

Hint

Track the first lock release and the transaction-completion boundary.

Answer B: Conflict serializability for well-formed transactions.

ADB12 explicitly guarantees conflict serializability for well-formed 2PL transactions. Locking without its phase discipline need not provide that ordering guarantee. Basic 2PL can still deadlock and may permit cascading aborts, so it does not guarantee successful completion. Nor does serializability require every read to observe the latest wall-clock update.

Why each choice is right or wrong

A - Incorrect: Basic 2PL can deadlock, so deadlock freedom is not its general guarantee.

B - Correct: The phase discipline guarantees conflict-serializable schedules for well-formed transactions.

C - Incorrect: Deadlock resolution or failures can still abort transactions.

D - Incorrect: Serializability does not require every read to see the newest wall-clock version.

Primary note proof

Study-note section ADB12

> Basic 2PL has a growing phase that acquires/upgrades locks and a shrinking phase that releases/downgrades them. Once any lock is released, no new lock may be acquired. The lock point is the instant the final lock is obtained. 2PL guarantees conflict serializability for well-formed transactions, but can deadlock and basic 2PL alone need not prevent cascading aborts.

Study-note section ADB12

> Two-phase locking orders access to data; two-phase commit coordinates a distributed commit decision.

Study-note section ADB12

> Strict 2PL retains exclusive locks until commit/abort.

Question sources

Advanced Database Systems review questions · Question 20

Question 154 | NS0076

What is a deadlock in database systems?

A. A transaction that executes too slowly

B. A circular wait where each transaction holds a resource another needs, and none can proceed

C. A type of hardware failure causing data loss

D. When a transaction violates the ACID consistency property

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer B: A circular wait among transactions holding resources needed by others.

ADB13 defines deadlock through a waiting cycle: each participant holds a resource another needs, so none can advance without an intervention. Slow work alone can still make progress and is not deadlock. Hardware loss and consistency violations are distinct problems that do not require a resource-wait cycle.

Why each choice is right or wrong

A - Incorrect: A slow transaction may still make progress and need not participate in a wait cycle.

B - Correct: Circular hold-and-wait prevents the involved transactions from progressing without intervention.

C - Incorrect: Hardware failure is a different failure category and does not define a lock wait cycle.

D - Incorrect: A consistency violation concerns invalid data states rather than necessarily circular waiting.

Primary note proof

Study-note section ADB13

> Deadlock requires a waiting cycle: T1 holds X and wants Y; T2 holds Y and wants X. In a wait-for graph, an edge Ti→Tj means Ti waits for a lock held by Tj. A cycle indicates deadlock under the ordinary lock-wait model. Abort a suitable victim, undo its effects and release its locks; repeat if other cycles remain. Aborting every deadlocked transaction also breaks cycles but wastes more work.

Study-note section ADB13

> A timeout aborts long waits but may mistake slow progress for deadlock.

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts.

Question sources

Advanced Database Systems review questions · Question 21

Question 155 | NS0077

In the Wait-Die deadlock prevention strategy, when a YOUNGER transaction needs a resource held by an OLDER transaction, the younger transaction:

A. Wounds (aborts) the older transaction

B. Waits for the older transaction to release the resource

C. Is immediately aborted and restarted

D. Receives the resource immediately due to higher priority

Hint

Identify requester and holder ages before applying the protocol's action.

Answer C: Abort the younger requester; retry with its original timestamp.

The requester is younger and the resource holder older. ADB14 says wait-die permits an older requester to wait but aborts a younger requester needing an older holder. C therefore gives the correct action. Restart should retain the original timestamp to improve relative priority; immediately aborted does not require an immediate successful retry or resource grant.

Why each choice is right or wrong

A - Incorrect: Wait-die does not let the younger requester abort the older holder.

B - Incorrect: In wait-die, the younger requester dies rather than waiting for the older holder.

C - Correct: The younger requester is aborted and may restart using its original timestamp.

D - Incorrect: The older holder retains priority; the younger requester does not receive the resource immediately.

Primary note proof

Study-note section ADB14

> A smaller timestamp means an older transaction. In wait-die, an older requester may wait for a younger holder; a younger requester needing an older holder aborts itself. In wound-wait, an older requester aborts the younger holder; a younger requester waits for the older holder. Always identify requester versus holder before applying the names.

Question sources

Advanced Database Systems review questions · Question 22

Question 156 | NS0078

In the Wound-Wait strategy, when an OLDER transaction needs a resource held by a YOUNGER transaction:

A. The older transaction waits for the younger

B. The older transaction wounds (aborts) the younger transaction

C. Both transactions are simultaneously aborted

D. A deadlock detection algorithm runs

Hint

Identify requester and holder ages before applying the protocol's action.

Answer B: The older requester aborts the younger holder.

ADB14 distinguishes wound-wait from wait-die: an older requester wounds the younger holder, while a younger requester waits for an older holder. Here the requester is explicitly older, so B applies. The rule prevents waiting cycles through age-based action; it does not require detecting an existing deadlock first or aborting both participants.

Why each choice is right or wrong

A - Incorrect: An older requester wounds the younger holder rather than waiting under wound-wait.

B - Correct: The older requester causes the younger resource holder to abort and release its lock.

C - Incorrect: The protocol aborts the younger holder, not both transactions.

D - Incorrect: This is a prevention action based on ages, not a requirement to run cycle detection first.

Primary note proof

Study-note section ADB14

> A smaller timestamp means an older transaction. In wait-die, an older requester may wait for a younger holder; a younger requester needing an older holder aborts itself. In wound-wait, an older requester aborts the younger holder; a younger requester waits for the older holder. Always identify requester versus holder before applying the names.

Study-note section ADB13

> A cycle indicates deadlock under the ordinary lock-wait model.

Study-note section ADB13

> A no-wait policy aborts immediately on an incompatible lock request and needs no age timestamp.

Question sources

Advanced Database Systems review questions · Question 23

Question 157 | NS0079

Optimistic concurrency control is BEST suited for environments where:

A. Transactions frequently write to the same data items

B. Write operations always conflict with read operations

C. Conflicts between transactions are rare (read-heavy workloads)

D. All transactions must complete in strict timestamp order

Hint

Consider conflicts, private work and validation costs, or lock-management overhead.

Answer C: Conflicts between transactions are rare (read-heavy workloads)

Optimistic concurrency control lets transactions read and compute privately, then validates overlapping read/write activity before publishing changes. When conflicts are rare, most work can commit without being discarded. Frequent competing writes or persistent read/write conflicts instead lead to costly retries. Strict timestamp completion describes a separate ordering constraint; it does not supply the low-conflict condition that makes optimistic execution suitable. Read-heavy activity is useful here only insofar as actual conflicts remain uncommon.

Why each choice is right or wrong

A - Incorrect: Incorrect: frequent competing writes cause validation failures and discarded work.

B - Incorrect: Incorrect: persistent read/write conflicts increase retry costs rather than favoring optimistic execution.

C - Correct: Correct: rare conflicts allow most private work to validate and commit.

D - Incorrect: Incorrect: a required completion order does not imply low contention; optimistic suitability depends on conflicts and retry cost.

Primary note proof

Study-note section ADB16

> Optimistic/validation control reads and computes privately, validates its read/write sets against overlapping transactions, then publishes if validation succeeds; otherwise it retries. It works well when conflicts are uncommon, but high contention wastes work on aborts.

Question sources

Advanced Database Systems review questions · Question 24

Question 158 | NS0080

Lock upgrade refers to changing a lock from:

A. Exclusive → Shared

B. Shared → Exclusive

C. No lock → Shared

D. Exclusive → No lock

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer B: Shared → Exclusive.

ADB11 explicitly names S→X an upgrade and X→S a downgrade. The stronger exclusive lock permits writing and can be granted only after conflicting other holders leave. Acquiring a new shared lock is initial acquisition rather than conversion, while releasing an exclusive lock is unlocking. Thus B alone names a lock upgrade.

Why each choice is right or wrong

A - Incorrect: Exclusive-to-shared weakens permissions and is a downgrade.

B - Correct: Shared-to-exclusive strengthens the held lock to permit writing, subject to compatibility.

C - Incorrect: No-lock-to-shared is initial acquisition, not conversion of an already held lock.

D - Incorrect: Exclusive-to-no-lock is release rather than upgrade.

Primary note proof

Study-note section ADB11

> A transaction must own the appropriate lock before access and can release only its own locks. S→X is an upgrade, possible only after conflicting other holders leave. X→S is a downgrade. The lock manager decides requests; the lock table stores granted locks and waiting requests. One transaction may request a lock on an already locked item: it is granted if compatible, otherwise waits or aborts under the policy. No double-locking is not a prohibition on shared readers.

Question sources

Advanced Database Systems review questions · Question 25

Question 159 | NS0081

Which trace demonstrates an incorrect-summary anomaly? The intended anomaly is a mixed-version aggregate, distinct from a nonrepeatable read.

A. Two transactions both delete the same record simultaneously

B. Initially A=B=100. During T2's transfer of 10 from A to B, T1 sums old A=100 and new B=110, returning 210 although the total-balance invariant is 200.

C. A transaction reads uncommitted data from another transaction

D. A transaction fails to acquire all necessary locks

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer B: B. Initially A=B=100. During T2's transfer of 10 from A to B, T1 sums old A=100 and new B=110, returning 210 although the total-balance invariant is 200.

The transfer preserves the total balance at 200, but T1 combines the pre-transfer debit balance with the post-transfer credit balance. Its computed 100+110=210 is not a valid summary of a consistent state. This is an incorrect-summary anomaly. Dirty reading is defined by an uncommitted dependency, and lock failure or simultaneous deletion alone does not establish this mixed aggregate.

Why each choice is right or wrong

A - Incorrect: Concurrent deletion alone does not specify an inconsistent aggregate.

B - Correct: The mixed old/new balances yield 210 instead of the invariant 200.

C - Incorrect: Reading another transaction's uncommitted value defines a dirty read.

D - Incorrect: Failure to acquire locks concerns scheduling rather than this aggregate result.

Primary note proof

Study-note section ADB08

> Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Study-note section ADB08

> Incorrect summary mixes mutually inconsistent versions: a transfer moves 10 from A=100 to B=100; a sum reading old A=100 and new B=110 gets 210 instead of the invariant 200.

Study-note section ADB08

> A single committed update from B=20 to B=50 does not alone prove an invalid summary of 90.

Question sources

Advanced Database Systems review questions · Question 26

Question 160 | NS0082

Starvation in a database system occurs when:

A. A transaction consumes excessive memory resources

B. A transaction is repeatedly bypassed by higher-priority transactions and never executes

C. A deadlock persists longer than the timeout period

D. The transaction log becomes completely full

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer B: Indefinite postponement while other transactions keep progressing.

ADB13 defines starvation as indefinite postponement while other work proceeds, without requiring any waiting cycle. Repeatedly favoring higher-priority work can prevent a transaction from executing, matching B. Memory consumption, a full log or a long deadlock are different resource/failure conditions; none alone defines the fairness failure of starvation.

Why each choice is right or wrong

A - Incorrect: Excessive memory use is a resource problem rather than the definition of indefinite postponement.

B - Correct: Repeated bypass by other transactions can indefinitely postpone this transaction while work continues.

C - Incorrect: Deadlock requires a waiting cycle; starvation can occur without one.

D - Incorrect: A full log can block work but does not itself define unfair indefinite postponement.

Primary note proof

Study-note section ADB13

> Deadlock requires a waiting cycle: T1 holds X and wants Y; T2 holds Y and wants X. In a wait-for graph, an edge Ti→Tj means Ti waits for a lock held by Tj. A cycle indicates deadlock under the ordinary lock-wait model. Abort a suitable victim, undo its effects and release its locks; repeat if other cycles remain. Aborting every deadlocked transaction also breaks cycles but wastes more work.

Study-note section ADB13

> Starvation is indefinite postponement while other work proceeds, without a required cycle.

Study-note section ADB13

> Fair queues, sensible victim selection and retaining original transaction timestamps on retry reduce starvation; deadlock freedom alone does not prove fairness.

Question sources

Advanced Database Systems review questions · Question 27

Question 161 | NS0083

Which concurrency control technique assigns a unique start-time value to each transaction and uses this to order execution and resolve conflicts?

A. Two-Phase Locking

B. Shadow Paging

C. Timestamp-Based Concurrency Control

D. Optimistic Concurrency Control

Hint

Compare the transaction timestamp with the item's recorded read and write timestamps.

Answer C: Timestamp-based concurrency control.

ADB15 assigns TS(T) to each transaction and compares it with item read_TS and write_TS values. Those tests reject operations that would violate timestamp order, so C matches the start-time ordering technique. 2PL orders locking, optimistic control validates private work, and shadow paging is a recovery/storage technique rather than this conflict-ordering protocol.

Why each choice is right or wrong

A - Incorrect: 2PL orders lock acquisition and release rather than defining the timestamp validity tests.

B - Incorrect: Shadow paging preserves page versions for recovery rather than ordering concurrent transaction operations by timestamp.

C - Correct: Timestamp ordering compares transaction timestamps with item read/write timestamps to resolve conflicts.

D - Incorrect: Optimistic control validates private read/write work rather than being defined by these timestamp-order tests.

Primary note proof

Study-note section ADB15

> Assign each transaction TS(T). For item X track read_TS(X), the largest timestamp of a successful reader, and write_TS(X), that of its latest successful writer. A read by T aborts if TS(T)<write_TS(X); otherwise it succeeds and updates read_TS. A write aborts if TS(T)<read_TS(X) or TS(T)<write_TS(X); otherwise it succeeds and sets write_TS. These tests preserve timestamp order.

Study-note section ADB16

> Multiversion control keeps versions so a reader can use an appropriate snapshot.

Study-note section ADB29

> Redo-only and NO-UNDO/REDO mean the same thing, so those cannot be separate unique answers.

Related reading (context, not primary note proof)

Study-note section ADB12

Question sources

Advanced Database Systems review questions · Question 28

Question 162 | NS0084

A well-formed transaction must follow which rule regarding locking?

A. Lock all items at the end of the transaction

B. Lock a data item BEFORE reading or writing it

C. Lock items only when modifying them, not when reading

D. Acquire all locks in alphabetical order by item name

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer B: Acquire the appropriate lock before accessing the item.

ADB11 requires ownership of an appropriate lock before access and permits releasing only one's own locks. Reads therefore need shared or suitable stronger locks in this basic model; writes need exclusive ownership. Alphabetical acquisition may be a separate prevention convention, but it is not the general well-formedness rule and locks cannot be postponed until after access.

Why each choice is right or wrong

A - Incorrect: Locking only at the end leaves the earlier accesses unprotected.

B - Correct: A well-formed transaction holds an appropriate lock before reading or writing an item.

C - Incorrect: Basic lock-based access also requires an appropriate read lock, not only locks for writes.

D - Incorrect: A fixed resource order can help prevention, but alphabetical order is not the general well-formedness definition.

Primary note proof

Study-note section ADB11

> A shared (S) lock permits reading; multiple S locks on the same item can coexist. An exclusive (X) lock permits its holder to read and write and conflicts with another S or X on that item. Locks on different items do not conflict merely because one is exclusive. These are the basic lock-based rules; multiversion systems can permit additional snapshot-reader behavior.

Study-note section ADB11

> A transaction must own the appropriate lock before access and can release only its own locks.

Study-note section ADB13

> Aborting every deadlocked transaction also breaks cycles but wastes more work.

Question sources

Advanced Database Systems review questions · Question 29

Question 163 | NS0085

Lock timeout as a deadlock solution works by:

A. Detecting cycles in the wait-for graph

B. Building a priority queue of all waiting transactions

C. Automatically aborting a transaction that has waited too long for a lock

D. Locking all required data items before execution begins

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer C: Abort a transaction after its lock wait exceeds the timeout.

ADB13 describes timeout as aborting long waits, while warning it may mistake slow progress for deadlock. C directly states that mechanism. Detecting cycles is a different method; acquiring all needed locks first is conservative locking; a priority queue alone neither defines a timeout nor guarantees elimination of waiting cycles.

Why each choice is right or wrong

A - Incorrect: Cycle detection is graph-based deadlock detection rather than a wait-duration timeout.

B - Incorrect: A priority queue alone does not enforce the stated wait-duration abort policy.

C - Correct: The policy aborts work whose lock wait exceeds the chosen threshold.

D - Incorrect: Obtaining all locks before execution is conservative locking, not timeout handling.

Primary note proof

Study-note section ADB13

> Deadlock requires a waiting cycle: T1 holds X and wants Y; T2 holds Y and wants X. In a wait-for graph, an edge Ti→Tj means Ti waits for a lock held by Tj. A cycle indicates deadlock under the ordinary lock-wait model. Abort a suitable victim, undo its effects and release its locks; repeat if other cycles remain. Aborting every deadlocked transaction also breaks cycles but wastes more work.

Study-note section ADB13

> A timeout aborts long waits but may mistake slow progress for deadlock.

Study-note section ADB12

> Conservative/static 2PL obtains all needed locks before execution and does not begin while only a subset is available, preventing hold-and-wait deadlock.

Question sources

Advanced Database Systems review questions · Question 30

Question 164 | NS0086

What are the three phases of query processing in the correct order?

A. Optimization, Parsing, Evaluation

B. Parsing & Translation, Optimization, Evaluation

C. Evaluation, Translation, Optimization

D. Parsing, Evaluation, Optimization

Hint

Separate query representation, strategy selection and execution.

Answer B: Parsing/translation → optimization → evaluation.

ADB17 checks grammar and semantics, constructs a logical representation, chooses an equivalent efficient execution plan and evaluates it. B groups parsing and translation into the first phase and preserves that dependency order. Plans cannot sensibly be evaluated before construction and selection, and optimization does not precede understanding the query's syntax and meaning.

Why each choice is right or wrong

A - Incorrect: Optimization cannot precede parsing and understanding the query in the stated pipeline.

B - Correct: It preserves analysis/translation, plan optimization and then execution/evaluation.

C - Incorrect: Evaluation cannot be the first phase before the query is represented and planned.

D - Incorrect: It executes before selecting the optimized plan, reversing optimization and evaluation.

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB17

> Query processing is the whole activity; selecting a suitable efficient strategy is optimization.

Study-note section ADB17

> DDL is not a mandatory stage before each SELECT, so authentication→DDL→DML permutations do not describe the processing sequence.

Question sources

Advanced Database Systems review questions · Question 31

Question 165 | NS0087

During query translation, SQL is typically converted into which internal representation for processing?

A. Machine code instructions

B. Binary decision trees

C. Relational algebra expressions

D. Compressed byte code

Hint

Separate query representation, strategy selection and execution.

Answer C: Relational algebra expressions or a logical query tree.

ADB17 states that translation constructs a logical representation, typically relational algebra or a logical query tree. This allows equivalent rewrites and plan choice before physical execution. Machine code or bytecode may appear in particular execution implementations, but they are not the general logical translation target asked for here; a binary decision tree is also not the defined representation.

Why each choice is right or wrong

A - Incorrect: Machine code is not the logical relational representation described for query translation.

B - Incorrect: A binary decision tree is not the note's general query-algebra representation.

C - Correct: Relational algebra expresses the logical operators that can be rewritten and optimized.

D - Incorrect: Compressed bytecode is not the stated general logical query representation.

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB17

> A query tree has base relations at leaves and operators at internal nodes; a query graph emphasizes relations and predicates.

Study-note section ADB17

> A physical execution plan adds scan/index choices and algorithms.

Question sources

Advanced Database Systems review questions · Question 32

Question 166 | NS0088

The query optimizer's primary goal is to:

A. Ensure queries return results in sorted order

B. Find an execution plan that minimizes the overall cost of retrieving data

C. Maximize the number of indexes used in the query

D. Ensure all tables are accessed in the order written in SQL

Hint

Identify the resource objective, counts and unit costs before comparing execution strategies.

Answer B: Choose a low-cost equivalent execution plan.

ADB19 estimates resource costs such as I/O, CPU, memory, temporary storage and communication for candidate plans. Optimization compares equivalent ways to return the same result and chooses a suitable low-cost strategy. It does not seek index count, force textual table order or impose sorting that the query never requested. The estimate need not prove a global real-world minimum.

Why each choice is right or wrong

A - Incorrect: Sorting is required only when query semantics need it, not as the universal optimizer goal.

B - Correct: The optimizer seeks an efficient equivalent plan under the selected resource-cost model.

C - Incorrect: More indexes used does not necessarily mean lower cost or better execution.

D - Incorrect: Equivalent join reordering can reduce cost; SQL textual order need not dictate access order.

Primary note proof

Study-note section ADB19

> A cost model estimates resources for a candidate plan: page I/O, CPU, memory, temporary storage and network traffic. Catalog statistics include row/page counts, distinct values and distributions. Selection with fraction s predicts s×N rows. For non-NULL equijoin keys, a common estimate is $|R|\times|S|/\max(V(R,k),V(S,k))$, assuming uniform values and containment of the smaller distinct-value domain within the larger. Adjust counts for NULLs; if either non-NULL input is empty, the result is zero. Skew, correlations or incomplete domain overlap can invalidate this estimate; known disjoint domains give zero matches.

Study-note section ADB23

Avoid universal rules that joins always beat subqueries, functions can never use indexes or more indexes always improve performance.

Study-note section ADB18

Cost-based optimization estimates candidate plans instead of relying on rules alone.

Question sources

Advanced Database Systems review questions · Question 33

Question 167 | NS0089

In the textbook local rotating-disk cost model, data is not memory-resident and query execution is disk-bound; client/network and CPU costs are negligible by comparison. Which component dominates the estimated query cost?

A. CPU processing time

B. Disk I/O time (seek time + transfer time)

C. Memory allocation overhead

D. Network latency to the client

Hint

Identify the resource objective, counts and unit costs before comparing execution strategies.

Answer B: B. Disk I/O time (seek time + transfer time)

The specified disk-bound model estimates the cost of seeks and transferred blocks. Because CPU, memory-management and network costs are negligible by assumption, disk I/O dominates this query's estimated cost. This is a contextual model rather than a universal fact about all databases: memory-resident or distributed workloads can have different dominant resources. The answer follows the scenario's stated objective.

Why each choice is right or wrong

A - Incorrect: CPU cost is negligible in the specified model.

B - Correct: Seek and transfer time dominate the stated disk-bound workload.

C - Incorrect: Memory-management overhead is negligible here.

D - Incorrect: Client/network cost is negligible in this local scenario.

Primary note proof

Study-note section ADB19

> For a simple rotating-disk model, cost = seeks×seek_time + transferred_blocks×block_transfer_time. Five seeks at 5 ms plus ten blocks at 1 ms cost 35 ms. Disk I/O often dominates textbook local-disk examples; CPU can dominate in-memory work and network can dominate distributed plans. State the assumed cost objective before ranking strategies.

Question sources

Advanced Database Systems review questions · Question 34

Question 168 | NS0090

In the disk I/O cost formula Cost = (S x tS) + (B x tT), what does 'B' represent?

A. Block size in bytes

B. Number of disk blocks transferred

C. Buffer pool size

D. Binary search operations

Hint

Identify the resource objective, counts and unit costs before comparing execution strategies.

Answer B: The number of transferred disk blocks.

ADB19 computes disk cost as seeks times seek duration plus transferred blocks times per-block transfer time. Therefore B in the formula is a block count, making B×tT the total transfer time. Block size, buffer capacity and search-operation count are different quantities and would not have the units required by this expression.

Why each choice is right or wrong

A - Incorrect: Block size is bytes per block rather than the number multiplied by per-block transfer time.

B - Correct: B counts transferred blocks, so B×tT gives their total transfer time.

C - Incorrect: Buffer-pool capacity is not the transferred-block count in this formula.

D - Incorrect: Binary-search operation count is not the disk-block transfer term.

Primary note proof

Study-note section ADB19

> For a simple rotating-disk model, cost = seeks×seek_time + transferred_blocks×block_transfer_time. Five seeks at 5 ms plus ten blocks at 1 ms cost 35 ms. Disk I/O often dominates textbook local-disk examples; CPU can dominate in-memory work and network can dominate distributed plans. State the assumed cost objective before ranking strategies.

Question sources

Advanced Database Systems review questions · Question 35

Question 169 | NS0091

The UNDO operation in database recovery is used to:

A. Re-apply changes of committed transactions not yet written to disk

B. Create new checkpoint entries in the log

C. Reverse changes made by incomplete or aborted transactions

D. Transfer data from cache to permanent disk storage

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer C: Reverse incomplete or aborted transaction effects.

ADB27 uses undo to restore before-images of incomplete effects, usually traversing a transaction’s updates backward. This removes changes that must not survive. Reapplying committed changes is redo, checkpoint records bound recovery searches, and transferring dirty buffers to storage is page flushing. Only C describes undo rather than another recovery or storage operation.

Why each choice is right or wrong

A - Incorrect: Reapplying missing committed updates is redo rather than undo.

B - Incorrect: Checkpoint creation records recovery state and is not reversal of incomplete effects.

C - Correct: Undo restores old values to remove incomplete or aborted transaction changes.

D - Incorrect: Writing a buffer to disk is flushing rather than restoring before-images.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history.

Study-note section ADB30

> In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo.

Related reading (context, not primary note proof)

Study-note section ADB07

Question sources

Advanced Database Systems review questions · Question 36

Question 170 | NS0092

The REDO operation in database recovery is used to:

A. Abort all active transactions found in the log

B. Reverse changes made by failed transactions

C. Re-apply changes of committed transactions not yet written to disk

D. Create shadow copies of database pages

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer C: Reapply required logged after-images, including committed changes missing from pages.

ADB27 describes redo as installing logged after-images when necessary. In the simple teaching model it preserves committed work not yet on disk, matching C. ARIES may repeat history more broadly before loser undo, so C describes the ordinary purpose rather than every algorithm's full phase. Aborting transactions and reversing failed changes are different actions.

Why each choice is right or wrong

A - Incorrect: Aborting active transactions is not the operation of applying logged after-images.

B - Incorrect: Reversing incomplete effects is undo rather than redo.

C - Correct: Redo can restore committed changes whose data pages were not durable at the crash.

D - Incorrect: Shadow-page creation is a separate copy-on-write technique rather than logged redo.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB29

> Shadow/copy-on-write update writes a new location while preserving the committed version.

Study-note section ADB29

> Shadow paging keeps an unchanged committed page map and a modified map; commit durably installs a new root/map pointer atomically after required pages are durable.

Question sources

Advanced Database Systems review questions · Question 37

Question 171 | NS0093

What is the fundamental durable-write ordering rule of write-ahead logging? In B, 'actual data changes' means writing the corresponding changed data page to persistent storage, not modifying a volatile buffer.

A. Write data to disk before creating the log record

B. Log records must be written to stable storage BEFORE making the actual data changes

C. All transactions must commit before any log entries are written

D. Log records are written only after a system crash occurs

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer B: B. Log records must be written to stable storage BEFORE making the actual data changes

WAL requires the recovery log for an update to reach stable storage before its corresponding changed page is written to disk. A volatile buffer may already contain the change, so this rule does not order every CPU assignment after a log flush. A reverses the required durable order; C and D postpone logs until too late to support recovery of earlier persistent changes.

Why each choice is right or wrong

A - Incorrect: Data-before-log reverses WAL's required durable ordering.

B - Correct: With the defined durable-write scope, the log reaches stable storage first.

C - Incorrect: Update logs may need to be durable before commit and before data flushing.

D - Incorrect: Recovery requires logs that were durable before the crash.

Primary note proof

Study-note section ADB27

> Write-ahead logging requires the update log needed for recovery to reach stable storage before the corresponding changed data page reaches disk. To acknowledge a durable commit, force the transaction's required log records, including its commit record, to stable storage. A buffer can be changed before that log flush; WAL orders durable writes, not every CPU assignment.

Study-note section ADB27

> Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not.

Study-note section ADB27

> Redo installs logged after-images when necessary, usually forward.

Question sources

Advanced Database Systems review questions · Question 38

Question 172 | NS0094

In a simple quiescent checkpoint exercise, all transactions have completed and the required log/data pages are flushed at the checkpoint. How does that checkpoint reduce later recovery search work?

A. Preventing all types of hardware failures

B. Encrypting log records for faster access

C. Providing a known consistent state so recovery can start from there instead of the log beginning

D. Automatically committing all active transactions periodically

Hint

Identify which history a recovery boundary can safely exclude and whether active work spans it.

Answer C: C. Providing a known consistent state so recovery can start from there instead of the log beginning

The quiescent checkpoint supplies a durable boundary for the simplified exercise. With no active transaction spanning it and the required pages flushed, recovery need not scan all earlier history to redo those completed changes. C captures that bounded search. Checkpointing does not prevent hardware failure, gain its benefit from encryption or automatically commit active transactions. Fuzzy checkpoints require different starting-point qualifications.

Why each choice is right or wrong

A - Incorrect: Recording recovery state does not prevent hardware failures.

B - Incorrect: Encryption is not the mechanism reducing log-search work.

C - Correct: The specified durable checkpoint bounds the history needed in this exercise.

D - Incorrect: Transactions have completed here; a checkpoint does not automatically commit active work.

Primary note proof

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history. In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo. Active transactions can span the checkpoint, so their relevant earlier updates can still need undo. Do not assume every transaction before a checkpoint needs no processing.

Study-note section ADB30

> Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Study-note section ADB30

> A checkpoint is neither a full backup nor an automatic commit of active transactions.

Question sources

Advanced Database Systems review questions · Question 39

Question 173 | NS0095

In deferred update (No-Undo) recovery, actual database modifications are written to disk:

A. Immediately as each SQL statement executes

B. Only AFTER the transaction successfully commits

C. When a checkpoint operation is triggered

D. At regular 5-minute intervals regardless of transaction state

Hint

Separate update timing from update location and follow the commit boundary.

Answer B: After successful commit, in the deferred-update model.

ADB29 keeps uncommitted changes outside the persistent database. Only committed effects may therefore be installed in its actual database pages, which is the property B expresses. This does not require every committed page to be flushed immediately. Checkpoints and periodic timers cannot independently authorize writing uncommitted changes while retaining the claimed No-Undo model.

Why each choice is right or wrong

A - Incorrect: Immediate per-statement persistent writes can expose uncommitted effects and do not define deferred update.

B - Correct: The model does not install uncommitted effects in the persistent database; committed changes can be written later.

C - Incorrect: A checkpoint alone does not make an uncommitted transaction eligible for persistent installation.

D - Incorrect: A fixed timer cannot ignore transaction commit status while preserving deferred-update semantics.

Primary note proof

Study-note section ADB29

> Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO. Redo-only and NO-UNDO/REDO mean the same thing, so those cannot be separate unique answers. Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Study-note section ADB29

> Shadow/copy-on-write update writes a new location while preserving the committed version.

Study-note section ADB29

> Shadow paging keeps an unchanged committed page map and a modified map; commit durably installs a new root/map pointer atomically after required pages are durable.

Question sources

Advanced Database Systems review questions · Question 40

Question 174 | NS0096

Shadow paging recovery technique preserves the original database pages by:

A. Creating a backup on a separate server

B. Writing all modifications to NEW shadow pages, keeping originals intact until commit

C. Encrypting pages before any modification

D. Storing pages in sorted order for faster access

Hint

Separate update timing from update location and follow the commit boundary.

Answer B: Write changed data to new pages while preserving the committed page map.

ADB29 preserves the committed version and writes changed data at new locations. Shadow paging keeps the old page map plus a modified map; commit atomically installs the new durable root after needed pages are durable. B captures the copy-on-write preservation, although the unchanged old map is the shadow rather than the name of each newly modified page.

Why each choice is right or wrong

A - Incorrect: A separate-server backup is not the page-map copy-on-write mechanism of shadow paging.

B - Correct: Changes use new pages, preserving the originals until the new committed map is safely installed.

C - Incorrect: Encryption does not preserve the old page mapping or provide the stated rollback mechanism.

D - Incorrect: Sorting pages is an access-layout choice rather than preserving committed copies.

Primary note proof

Study-note section ADB29

> In-place update overwrites the original location. Shadow/copy-on-write update writes a new location while preserving the committed version. Shadow paging keeps an unchanged committed page map and a modified map; commit durably installs a new root/map pointer atomically after required pages are durable. Before commit, a crash uses the old map. It is an

alternative to conventional log-based recovery, with allocation, fragmentation and concurrency costs; it is not primarily a special action done only during recovery.

Study-note section ADB26

A full backup copies the database to separate storage; scheduled backups plus archived logs support restoration to a chosen recoverable point.

Study-note section ADB26

Rollback handles failed/uncommitted work; redo preserves committed work missing from pages; checkpoints reduce log-search work; backups provide another stored copy.

Related reading (context, not primary note proof)

Study-note section ADB24

Question sources

Advanced Database Systems review questions · Question 41

Question 175 | NS0097

Which type of database failure is caused by accidental deletion of important data by a user?

A. Hardware failure

B. Natural disaster

C. Software error

D. Human error

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer D: Human error.

The described cause is an accidental user action, not faulty hardware, a natural event or a software defect. ADB26 treats human mistakes separately and notes they may need point-in-time restoration before a bad committed operation. Ordinary crash redo would otherwise preserve a committed deletion rather than recognize the user’s intention.

Why each choice is right or wrong

A - Incorrect: No hardware malfunction is stated; the cause is the user's mistaken deletion.

B - Incorrect: A natural disaster is not the cause described in the prompt.

C - Incorrect: The software may have executed the requested deletion correctly; the stated mistake belongs to the user.

D - Correct: Accidental deletion by a user is a human error.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB-S05

> An accidental user deletion is human error even if the software executes the requested command correctly.

Study-note section ADB-S05

> A software failure originates in erroneous program behavior, such as a logical programming defect, invalid operation or operating-system fault.

Question sources

Advanced Database Systems review questions · Question 42

Question 176 | NS0098

The transaction log records 'before and after values' for data modifications primarily to support:

A. Query optimization

B. User access control

C. UNDO (restore old value) and REDO (apply new value) operations during recovery

D. Database compression and storage efficiency

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer C: Undo with old values and redo with logged new values.

ADB27 restores before-images when undoing incomplete work and installs after-images when redoing logged effects. Recording both values therefore supports reversal and reapplication after failure. Query costing uses catalog statistics, access control uses privileges, and compression is a different storage concern; none explains the specific recovery value pair as directly.

Why each choice is right or wrong

A - Incorrect: Query optimization estimates plan cost rather than restoring old/new transaction values.

B - Incorrect: Access control checks privileges rather than using before/after images to authorize users.

C - Correct: Old values support reversal and new values support necessary reapplication during recovery.

D - Incorrect: Recording two values can add storage overhead and is not primarily a compression mechanism.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB19

> A cost model estimates resources for a candidate plan: page I/O, CPU, memory, temporary storage and network traffic.

Study-note section ADB19

> Catalog statistics include row/page counts, distinct values and distributions.

Related reading (context, not primary note proof)

Study-note section ADB24

Question sources

Advanced Database Systems review questions · Question 43

Question 177 | NS0099

Which type of distributed database uses the SAME DBMS software at every site?

A. Heterogeneous distributed database

B. Homogeneous distributed database

C. Middleware-based database

D. Partially replicated database

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer B: Homogeneous distributed database.

ADB33 identifies homogeneous systems by compatible DBMS technology across sites. Using the same DBMS software fits that classification. Heterogeneous systems integrate differing engines, models or interfaces. Middleware describes integration architecture and partial replication describes placement of copies, so neither substitutes for the homogeneity classification asked for here.

Why each choice is right or wrong

A - Incorrect: Heterogeneous systems integrate differing database technologies rather than the same DBMS throughout.

B - Correct: Using the same DBMS at each site fits homogeneous distributed database technology.

C - Incorrect: Middleware describes communication/integration responsibilities, not whether DBMS technology is the same.

D - Incorrect: Partial replication describes where selected copies reside rather than which DBMS software sites use.

Primary note proof

Study-note section ADB33

> A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Study-note section ADB34

> These describe communication responsibilities, not fragmentation types.

Question sources

Advanced Database Systems review questions · Question 44

Question 178 | NS0100

Horizontal fragmentation divides a table based on:

A. Splitting columns into separate subsets

B. Splitting rows into subsets based on a condition

C. Combining two tables into one larger table

D. Replicating the table to multiple sites

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer B: Split rows by predicates.

ADB36 defines horizontal fragmentation as selecting row subsets using predicates and reconstructing the relation by union. Column subsets belong to vertical fragmentation. Combining tables is not fragmentation, and copying a whole table is replication. Thus B identifies both the item being split and the rule used to decide fragment membership.

Why each choice is right or wrong

A - Incorrect: Column splitting is vertical fragmentation rather than horizontal fragmentation.

B - Correct: Horizontal fragments contain rows selected by stated predicates.

C - Incorrect: Combining tables increases the combined relation rather than splitting one table into row subsets.

D - Incorrect: Replicating copies preserves the whole table at more sites rather than partitioning its rows.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Study-note section ADB37

> Full replication does not inherently mean every copy is updated simultaneously; its synchronization rule must be stated.

Question sources

Advanced Database Systems review questions · Question 45

Question 179 | NS0101

In vertical fragmentation, why must the PRIMARY KEY be included in every fragment?

A. To speed up INSERT operations

B. To prevent duplicate rows

C. To enable reconstruction of the original table by joining fragments on the primary key

D. To ensure proper replication across sites

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer C: A repeated key or row identifier lets vertical pieces join back losslessly.

ADB36 repeats a key or row identifier across vertical fragments so attributes from the same original row can be matched during reconstruction. C states that purpose. The general requirement is a suitable identifier, not necessarily the particular declared primary key if another row identifier is used. It is unrelated to mandatory replication or inherently faster inserts.

Why each choice is right or wrong

A - Incorrect: Faster insertion is not why vertical fragments share a row identifier.

B - Incorrect: Avoiding duplicate values is not sufficient to align column pieces from the same original row.

C - Correct: The shared key identifies matching original rows when the column fragments are joined.

D - Incorrect: Replication copies data and is distinct from reconstructing vertical column fragments.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB36

> Hybrid fragmentation combines row and column splits in either suitable order.

Study-note section ADB36

> Reconstruct by reversing the operations: key joins for corresponding vertical pieces and unions for horizontal pieces.

Question sources

Advanced Database Systems review questions · Question 46

Question 180 | NS0102

Which data replication scheme stores each data fragment at ONLY ONE site with NO copies?

A. Full replication

B. Partial replication

C. No replication (non-redundant allocation)

D. Hybrid replication

Hint

Separate copy placement, propagation timing and update-maintenance cost.

Answer C: No replication, or nonredundant allocation.

ADB37 explicitly calls storing each fragment once nonredundant allocation. Full replication places the whole database at every relevant site, while partial replication copies selected fragments to selected sites. A hybrid can mix strategies, but the stem specifies exactly one location and no copies for every fragment, so C is the precise classification.

Why each choice is right or wrong

A - Incorrect: Full replication stores copies throughout the relevant sites rather than one copy per fragment.

B - Incorrect: Partial replication still copies selected fragments and therefore does not mean no copies.

C - Correct: Nonredundant allocation stores each fragment at one site without replicas.

D - Incorrect: A hybrid can combine policies, but the stated universal no-copy arrangement is specifically nonreplicated.

Primary note proof

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once. Copies improve read locality and can improve availability but require storage and update propagation. They can also spread mistakes, so replication does not replace independent backups.

Study-note section ADB37

> Full replication does not inherently mean every copy is updated simultaneously; its synchronization rule must be stated.

Study-note section ADB37

> Snapshot replication transfers a copy at intervals; incremental replication transfers changes.

Question sources

Advanced Database Systems review questions · Question 47

Question 181 | NS0103

Transparency in distributed databases means that:

A. All data is encrypted and transparent to hackers

B. The distribution complexity is hidden from users who interact as if using a single database

C. Users can see all technical details of how data is distributed

D. All sites must use the same operating system

Hint

Identify exactly which detail is hidden from the user.

Answer B: Hide distribution details behind a single logical database experience.

ADB35 defines transparency through which distribution details are hidden: sites, fragment reconstruction or replica selection/maintenance. Together these can let clients work with a single logical database abstraction. B captures that broad meaning. Transparency is not encryption, disclosure of every technical detail or a requirement that operating systems be identical.

Why each choice is right or wrong

A - Incorrect: Encryption protects content; transparency here concerns hiding distribution complexity from legitimate users.

B - Correct: Distribution details can be hidden so users work with one logical database abstraction.

C - Incorrect: Transparency hides selected implementation details rather than exposing all of them.

D - Incorrect: Operating-system uniformity does not define distribution transparency.

Primary note proof

Study-note section ADB35

> Location transparency hides which site holds a named object. Fragmentation transparency hides the existence and reconstruction of fragments: users query the whole logical relation. Replication transparency hides which copy is used and how copies are maintained. The broad single-logical-database experience may involve all three.

Study-note section ADB33

> Different operating systems are possible but do not define the distinction.

Study-note section ADB24

> Encryption protects stored or transmitted content; integrity constraints restrict valid data; auditing records actions.

Question sources

Advanced Database Systems review questions · Question 48

Question 182 | NS0104

Hybrid fragmentation combines which two fragmentation types?

A. Replication and Partitioning

B. Horizontal and Vertical fragmentation

C. Client-Server and Middleware architectures

D. Homogeneous and Heterogeneous distribution

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer B: Horizontal and vertical fragmentation.

ADB36 defines hybrid fragmentation as combining row and column splits in a suitable order. Reconstructing reverses those operations using unions for horizontal pieces and key joins for corresponding vertical pieces. Replication, deployment architectures and DBMS homogeneity are different design dimensions rather than the two fragmentation types combined by hybrid fragmentation.

Why each choice is right or wrong

A - Incorrect: Replication copies data and is not the second fragmentation dimension in the hybrid definition.

B - Correct: Hybrid fragmentation combines horizontal row splits with vertical column splits.

C - Incorrect: Client-server and middleware classify architectures rather than fragmentation operations.

D - Incorrect: Homogeneous/heterogeneous classifies system technology compatibility rather than row/column splitting.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB36

> Hybrid fragmentation combines row and column splits in either suitable order.

Study-note section ADB36

> Reconstruct by reversing the operations: key joins for corresponding vertical pieces and unions for horizontal pieces.

Related reading (context, not primary note proof)

Study-note section ADB37

Study-note section ADB34

Study-note section ADB33

Question sources

Advanced Database Systems review questions · Question 49

Question 183 | NS0105

With full replication and synchronous propagation before acknowledgment, which listed disadvantage specifically concerns maintaining copies during an update-heavy workload? Assume usable replicas can serve reads.

A. Data becomes unavailable when any site fails

B. Updates incur propagation and coordination overhead because changes must reach the required copies before acknowledgment.

C. It requires all sites to use the same DBMS software

D. Queries can only be answered from the primary site

Hint

Separate copy placement, propagation timing and update-maintenance cost.

Answer B: B. Updates incur propagation and coordination overhead because changes must reach the required copies before acknowledgment.

Full replication stores the entire database at each relevant site, so updates create propagation and coordination work. The stated synchronous policy places that work before acknowledgment, adding update latency in the update-heavy scenario. This is the disadvantage described by B. Usable copies can serve reads and improve availability; full replication neither implies identical DBMS software nor restricts queries to one primary site.

Why each choice is right or wrong

A - Incorrect: Usable replicas can keep data available after one site fails.

B - Correct: The stated synchronous policy adds propagation and coordination work before acknowledgment.

C - Incorrect: Full replication is a copy-placement policy, not a requirement to standardize every DBMS.

D - Incorrect: Usable replicas can serve reads in the stated scenario.

Primary note proof

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once. Copies improve read locality and can improve availability but require storage and update propagation. They can also spread mistakes, so replication does not replace independent backups.

Study-note section ADB37

> Eager/synchronous replication coordinates propagation before the specified commit/acknowledgment point, increasing update latency.

Study-note section ADB37

> Full replication does not inherently mean every copy is updated simultaneously; its synchronization rule must be stated.

Question sources

Advanced Database Systems review questions · Question 50

Question 184 | NS0106

What does ACID stand for in database transactions?

A. Atomicity, Concurrency, Isolation, Durability

B. Atomicity, Consistency, Isolation, Durability

C. Availability, Consistency, Integrity, Durability

D. Atomicity, Correctness, Isolation, Distribution

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: Atomicity, Consistency, Isolation, Durability.

ADB06 gives all-or-none effects, preservation of declared rules, controlled interference and persistence after failure as the four ACID properties. B names them correctly. The other options substitute concurrency, availability, integrity, correctness or distribution for the defined terms, changing the acronym rather than describing an equivalent expansion.

Why each choice is right or wrong

A - Incorrect: Concurrency incorrectly replaces Consistency in the acronym.

B - Correct: The expansion exactly matches the four ACID property names.

C - Incorrect: Availability and Integrity replace Atomicity and Isolation, which are the defined terms.

D - Incorrect: Correctness and Distribution do not replace Consistency and Durability in ACID.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Question sources

Advanced Database Systems review questions · Question 1

Question 185 | NS0107

Which ACID property ensures that either ALL operations of a transaction complete or NONE do?

A. Consistency

B. Isolation

C. Atomicity

D. Durability

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer C: Atomicity.

The requested property concerns whether partial transaction effects can survive. ADB06 assigns all-or-none application to atomicity. A correct transaction also preserves consistency, concurrency needs isolation and completed commits need durability, but those are different guarantees. C is therefore the property matching the explicit ALL or NONE condition.

Why each choice is right or wrong

A - Incorrect: Consistency concerns preservation of integrity rules rather than the indivisible effect boundary itself.

B - Incorrect: Isolation governs concurrent interference rather than all-or-none transaction effects.

C - Correct: Atomicity makes the transaction effects succeed together or be removed together.

D - Incorrect: Durability protects committed effects rather than defining partial-success handling.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Question sources

Advanced Database Systems review questions · Question 2

Question 186 | NS0108

If A=5000 and B=8000, and 750 is transferred from A to B, what values satisfy the Consistency property?

A. A=4250, B=8750, sum=13000

B. A=4250, B=8000, sum=12250

C. A=5000, B=8750, sum=13750

D. A=5000, B=8000, sum=13000

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer A: A=4250, B=8750, total=13000.

Subtracting 750 from A gives 4250 and adding it to B gives 8750; their total remains 13000. A implements both the specified operation and its total-balance invariant. B and C apply only one side. D preserves the initial total but does not perform the requested transfer, so invariant preservation alone does not make it the successful-transfer result.

Why each choice is right or wrong

A - Correct: It performs the 750 debit and credit and preserves the total of 13000.

B - Incorrect: The debit occurs without its credit, reducing the total to 12250.

C - Incorrect: The credit occurs without its debit, increasing the total to 13750.

D - Incorrect: It preserves the starting total but leaves both accounts unchanged rather than performing the transfer.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Study-note section ADB06

> Unchanged or reverse-transfer balances can also preserve that sum; only the requested debit/credit implements the specified successful transfer.

Study-note section ADB06

> Both balances total 13,000.

Question sources

Advanced Database Systems review questions · Question 3

Question 187 | NS0109

Which ACID property ensures concurrent transactions do not interfere with each other?

A. Atomicity

B. Durability

C. Consistency

D. Isolation

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer D: Isolation.

ADB06 assigns concurrent-interference control to isolation. That is the distinguishing phrase in the stem. Atomicity handles partial effects, durability handles committed persistence, and consistency means correct work preserves declared rules. Different isolation levels provide different detailed guarantees, so this classification does not imply that all implementations offer serializability by default.

Why each choice is right or wrong

A - Incorrect: Atomicity governs partial effects within a transaction rather than concurrent interference.

B - Incorrect: Durability concerns persistence of committed changes after failures.

C - Incorrect: Consistency concerns declared integrity rules under correct business logic.

D - Correct: Isolation controls interference among concurrently executing transactions.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Question sources

Advanced Database Systems review questions · Question 4

Question 188 | NS0110

Which ACID property guarantees committed changes survive system crashes?

A. Atomicity

B. Consistency

C. Isolation

D. Durability

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer D: Durability.

ADB06 defines durability by persistence of committed effects despite failure. Recovery information can provide this even when changed data pages were buffered at the instant of commit. All-or-none application, invariant preservation and concurrent isolation serve different transaction guarantees, so only durability matches survival of successful commits after a crash.

Why each choice is right or wrong

A - Incorrect: Atomicity concerns all-or-none application rather than the persistence of completed commits.

B - Incorrect: Consistency concerns validity of declared invariants rather than crash survival of commits.

C - Incorrect: Isolation controls concurrent interactions rather than persistence after a crash.

D - Correct: Durability preserves committed effects across system failures.

Primary note proof

Study-note section ADB06

> A transaction is a logical unit of database work, including read-only work. Atomicity means all effects or none; consistency means correct work preserves integrity rules; isolation controls concurrent interference; durability preserves committed effects despite failure. A DBMS cannot automatically make incorrect business logic consistent.

Question sources

Advanced Database Systems review questions · Question 5

Question 189 | NS0111

A transaction that has executed all operations but NOT YET committed is in which state?

A. Active

B. Partially Committed

C. Committed

D. Failed

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: Partially committed.

ADB06 places a transaction in Partially committed after its last operation finishes but before durable commitment is assured. The stem states exactly that boundary. Active means operations are still executing; Committed needs successful commit; Failed needs an error. Finishing the operation list alone does not establish the committed state.

Why each choice is right or wrong

A - Incorrect: Active describes ongoing operation execution, not the interval after the last operation finishes.

B - Correct: The last operation finished, but commitment is not yet assured.

C - Incorrect: The stem explicitly says the transaction has not yet committed.

D - Incorrect: No execution or commit failure is reported.

Primary note proof

Study-note section ADB06

> Active means operations are executing. Partially committed means the last operation finished but durable commitment is not yet assured. Committed means commitment succeeded. A failure leads to Failed, then Aborted after rollback; a transaction may terminate or be restarted as a new attempt. Concurrency control primarily enforces isolation; recovery supports atomicity and durability. Transaction management is the broader subsystem, so both it and recovery management can be legitimate answers to broad responsibility questions.

Question sources

Advanced Database Systems review questions · Question 6

Question 190 | NS0112

In the Aborted state, what happens to all changes made by the transaction?

A. Changes are permanently saved

B. All changes are rolled back restoring the database to its pre-transaction state

C. The transaction waits for resources

D. The transaction restarts automatically

Hint

Track the transaction's guarantee or lifecycle boundary described by the scenario.

Answer B: Its effects have been rolled back; other committed transactions remain.

ADB06 reaches Aborted after rollback. Thus the aborted transaction’s own changes are removed rather than permanently saved. Restoring pre-transaction state must be understood relative to its effects, not erasing unrelated concurrent commits. A restart can be a new attempt, but is not automatic merely because the aborted state was reached.

Why each choice is right or wrong

A - Incorrect: Permanent saving is a committed outcome, not an aborted outcome.

B - Correct: Rollback removes this transaction's effects while retaining other transactions' legitimate committed work.

C - Incorrect: Waiting for a lock describes a blocked executing transaction rather than completed rollback.

D - Incorrect: An aborted transaction may be retried, but automatic restart is not guaranteed.

Primary note proof

Study-note section ADB06

> Active means operations are executing. Partially committed means the last operation finished but durable commitment is not yet assured. Committed means commitment succeeded. A failure leads to Failed, then Aborted after rollback; a transaction may terminate or be restarted as a new attempt. Concurrency control primarily enforces isolation; recovery supports atomicity and durability. Transaction management is the broader subsystem, so both it and recovery management can be legitimate answers to broad responsibility questions.

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward.

Study-note section ADB27

> Write-ahead logging requires the update log needed for recovery to reach stable storage before the corresponding changed data page reaches disk.

Question sources

Advanced Database Systems review questions · Question 7

Question 191 | NS0113

Which log record signals successful completion of transaction T?

A. [start_transaction, T]

B. [abort, T]

C. [write_item, T, X, old, new]

D. [commit, T]

Hint

Trace movement between program variables, buffers and disk, or the meaning of a recorded transaction event.

Answer D: [commit, T].

ADB07 identifies COMMIT as the record of successful transaction completion. START begins work, UPDATE records a data change and ABORT marks failed termination/rollback handling. A data write by itself does not prove the enclosing transaction committed, and an abort record can begin rollback rather than confirm it has completed.

Why each choice is right or wrong

A - Incorrect: The start record indicates beginning work rather than successful completion.

B - Incorrect: Abort records failure/rollback handling instead of successful commitment.

C - Incorrect: An update record describes one changed item, not successful transaction completion.

D - Correct: The commit record signals successful completion of transaction T.

Primary note proof

Study-note section ADB07

> A conceptual log can contain START, UPDATE(T,X,old,new), COMMIT, ABORT and END records. Before-images support undo; after-images support redo. Real systems may use physical or logical records, and need not log every read. COMMIT records successful completion; an ABORT record can mark the start of rollback rather than prove every undo step is complete.

Question sources

Advanced Database Systems review questions · Question 8

Question 192 | NS0114

The Read_item(X) command transfers data in which order?

A. Disk -> Program variable -> Buffer

B. Buffer -> Disk -> Program variable

C. Disk -> Buffer -> Program variable

D. Program variable -> Buffer -> Disk

Hint

Trace movement between program variables, buffers and disk, or the meaning of a recorded transaction event.

Answer C: Disk page → buffer → program variable.

ADB07 locates X's page, loads it into a buffer if absent and copies the item from that buffer into the program variable. C gives that conceptual data movement order. A cached page may avoid the physical disk read, but the program still reads through the buffer. The reverse direction describes a write path rather than read_item.

Why each choice is right or wrong

A - Incorrect: The page is buffered before the item is copied to the program variable.

B - Incorrect: A read does not first send the buffer to disk before obtaining the variable value.

C - Correct: The page enters the buffer if needed, then its item is copied into the program variable.

D - Incorrect: Program-variable-to-buffer-to-disk describes writing, not reading into a variable.

Primary note proof

Study-note section ADB07

> read_item(X) locates X's page, brings it into a buffer if absent, and copies X into a program variable. write_item(X) copies the new program value into the buffered page, marking it dirty. Writing that page to disk can occur later according to the buffer policy; write_item is not inherently an immediate disk flush. Insert and delete also modify data; sorting is a query-processing operation rather than a primitive item update.

Study-note section ADB07

> Real systems may use physical or logical records, and need not log every read.

Study-note section ADB07

> Before-images support undo; after-images support redo.

Question sources

Advanced Database Systems review questions · Question 9

Question 193 | NS0115

In a multi-user system, which mechanisms does the DBMS use to manage concurrent access?

A. Only backup systems

B. Locks, timestamps, and scheduling algorithms

C. Physical user separation

D. Only file system permissions

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer B: Locks, timestamp rules and transaction scheduling.

ADB11 defines compatible shared/exclusive locks and ADB15 orders conflicting operations with timestamp tests. These are actual concurrency-control mechanisms that constrain transaction schedules. Backups provide copies for recovery, and file permissions determine access rather than safe interleavings. Physical separation of people also does not coordinate their concurrent database operations.

Why each choice is right or wrong

A - Incorrect: Backups support restoration rather than scheduling safe concurrent reads and writes.

B - Correct: Locks and timestamp rules control conflicting accesses and permissible schedules.

C - Incorrect: Separating users physically does not coordinate their concurrent operations on shared database items.

D - Incorrect: File permissions alone do not enforce isolation among authorized concurrent transactions.

Primary note proof

Study-note section ADB11

> A shared (S) lock permits reading; multiple S locks on the same item can coexist. An exclusive (X) lock permits its holder to read and write and conflicts with another S or X on that item. Locks on different items do not conflict merely because one is exclusive. These are the basic lock-based rules; multiversion systems can permit additional snapshot-reader behavior.

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts.

Study-note section ADB09

> Two operations conflict when they belong to different transactions, access the same item and at least one writes.

Related reading (context, not primary note proof)

Study-note section ADB15

Study-note section ADB24

Question sources

Advanced Database Systems review questions · Question 10

Question 194 | NS0116

The Lost Update problem is also called:

A. Read-Read Problem

B. Write-Write Problem

C. Read-Write Problem

D. Phantom Read Problem

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer B: Write-write problem, as shorthand for a lost-update overwrite.

ADB08 illustrates both transactions reading 500, one writing 700 and the other later overwriting it with stale 600. The conflict is between their writes, which explains the offered write-write shorthand. This is not a claim that every pair of writes loses an update: the stale calculation and overwrite are essential. Read/read and phantom behavior describe different situations.

Why each choice is right or wrong

A - Incorrect: Two reads alone do not overwrite and lose an update.

B - Correct: The stale write overwrites another write, motivating this write-write shorthand.

C - Incorrect: The defining loss is the later stale overwrite, not merely an arbitrary read/write pair.

D - Incorrect: A phantom changes a predicate query's qualifying row set rather than overwriting one balance.

Primary note proof

Study-note section ADB08

> Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Question sources

Advanced Database Systems review questions · Question 11

Question 195 | NS0117

A Dirty Read occurs when:

A. A transaction reads only committed data

B. A transaction reads data modified by another uncommitted transaction

C. Two transactions read the same record simultaneously

D. A transaction writes invalid format data

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer B: Reading another transaction’s uncommitted data.

ADB08 defines dirty read through another transaction’s uncommitted write. The value can disappear if its writer aborts, leaving dependent work based on invalid data. Reading committed data or two readers accessing one item is not inherently dirty. Invalid-format writes concern data validity rather than whether the read-from transaction has committed.

Why each choice is right or wrong

A - Incorrect: Reading only committed values avoids the defining uncommitted-writer dependency.

B - Correct: The read consumes another transaction's modification before that writer commits.

C - Incorrect: Multiple readers alone do not create an uncommitted data dependency.

D - Incorrect: Invalid format concerns validation or integrity rather than the definition of a dirty read.

Primary note proof

Study-note section ADB08

> Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Question sources

Advanced Database Systems review questions · Question 12

Question 196 | NS0118

Which trace demonstrates an incorrect-summary anomaly, meaning an aggregate that mixes mutually inconsistent versions? Treat nonrepeatable read as a separate anomaly.

A. A transaction reads data after another commits

B. Initially A=B=100. T2 transfers 10 from A to B while T1 reads old A=100 and new B=110, reporting 210 rather than the invariant total 200.

C. Two transactions delete the same record

D. A transaction reads its own uncommitted data

Hint

Trace which transaction wrote each value and whether an overwrite or mixed-version read occurs.

Answer B: B. Initially A=B=100. T2 transfers 10 from A to B while T1 reads old A=100 and new B=110, reporting 210 rather than the invariant total 200.

The transfer subtracts and adds the same amount, preserving a total of 200. T1 instead adds an old A balance to a new B balance and obtains 210, combining versions that do not form a consistent summary. That is the incorrect-summary anomaly. Reading committed data, deleting a record or reading one's own uncommitted changes alone does not establish this aggregate defect.

Why each choice is right or wrong

A - Incorrect: A read after another commit is not by itself an inconsistent aggregate.

B - Correct: 100+110=210 mixes versions and violates the preserved total 200.

C - Incorrect: Concurrent deletions alone do not show this summary anomaly.

D - Incorrect: Reading one's own changes is not the stated mixed-version aggregate.

Primary note proof

Study-note section ADB08

> Lost update: T1 and T2 both read 500; T2 adds 200 and writes 700; T1 adds 100 to its stale value and writes 600. Correct serial execution yields 800. Dirty read: T2 reads a value written by uncommitted T1; if T1 aborts, dependent work must be reconsidered. Reading one's own changes is not a dirty read.

Study-note section ADB08

> Incorrect summary mixes mutually inconsistent versions: a transfer moves 10 from A=100 to B=100; a sum reading old A=100 and new B=110 gets 210 instead of the invariant 200.

Study-note section ADB08

> A single committed update from B=20 to B=50 does not alone prove an invalid summary of 90.

Question sources

Advanced Database Systems review questions · Question 13

Question 197 | NS0119

A Shared Lock allows:

A. Only one transaction to read an item

B. Multiple transactions to read simultaneously but not write

C. One transaction to read and write exclusively

D. Multiple transactions to write simultaneously

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer B: Compatible readers, with no writing under a shared lock.

ADB11 permits multiple shared locks on one item, each for reading. Writing requires a suitable exclusive lock, which conflicts with other shared holders. Thus B describes shared-lock behavior in the basic lock-based model. Exclusive read/write ownership and multiple simultaneous writers are not shared locking; multiversion snapshot rules are a separate model.

Why each choice is right or wrong

A - Incorrect: More than one transaction may hold compatible shared read locks on the same item.

B - Correct: Shared locks permit multiple readers but do not authorize writes under that lock mode.

C - Incorrect: Sole read/write ownership describes an exclusive lock.

D - Incorrect: Shared locks do not allow multiple transactions to write simultaneously.

Primary note proof

Study-note section ADB11

> A shared (S) lock permits reading; multiple S locks on the same item can coexist. An exclusive (X) lock permits its holder to read and write and conflicts with another S or X on that item. Locks on different items do not conflict merely because one is exclusive. These are the basic lock-based rules; multiversion systems can permit additional snapshot-reader behavior.

Study-note section ADB11

> One transaction may request a lock on an already locked item: it is granted if compatible, otherwise waits or aborts under the policy.

Study-note section ADB11

> No double-locking is not a prohibition on shared readers.

Question sources

Advanced Database Systems review questions · Question 14

Question 198 | NS0120

An Exclusive Lock allows:

A. Multiple transactions to read concurrently

B. Multiple transactions to write simultaneously

C. Only one transaction to both read and write, blocking all other access

D. Multiple readers and one writer simultaneously

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer C: The exclusive holder may read and write; conflicting access to that item is blocked.

ADB11 permits an exclusive lock holder to read and write and makes that item's X lock conflict with another S or X. C is correct in this basic model. Blocking all other access means conflicting access to the same item, not the entire database; locks on other items remain independent and multiversion systems can have additional snapshot-reader behavior.

Why each choice is right or wrong

A - Incorrect: Other transactions' shared reads conflict with an exclusive lock on the same item in this model.

B - Incorrect: Two exclusive writers cannot hold compatible locks on the same item.

C - Correct: One holder can read/write the locked item while incompatible same-item accesses wait or abort.

D - Incorrect: Concurrent same-item shared readers conflict with this exclusive writer under the basic rules.

Primary note proof

Study-note section ADB11

> A shared (S) lock permits reading; multiple S locks on the same item can coexist. An exclusive (X) lock permits its holder to read and write and conflicts with another S or X on that item. Locks on different items do not conflict merely because one is exclusive. These are the basic lock-based rules; multiversion systems can permit additional snapshot-reader behavior.

Study-note section ADB11

> One transaction may request a lock on an already locked item: it is granted if compatible, otherwise waits or aborts under the policy.

Study-note section ADB11

> S→X is an upgrade, possible only after conflicting other holders leave.

Question sources

Advanced Database Systems review questions · Question 15

Question 199 | NS0121

In Two-Phase Locking (2PL), during the Growing Phase a transaction:

A. Releases locks only

B. Acquires locks without releasing any

C. Acquires and releases locks freely

D. Makes no lock requests

Hint

Track the first lock release and the transaction-completion boundary.

Answer B: Acquire or upgrade locks without releasing them.

ADB12 defines growing as the acquisition/upgrade phase and shrinking as the release/downgrade phase. Once any lock is released, no further acquisition is allowed. B therefore matches growing. Freely mixing acquisition and release would abandon the two-phase discipline that guarantees conflict serializability for well-formed transactions.

Why each choice is right or wrong

A - Incorrect: Releasing locks belongs to shrinking rather than growing.

B - Correct: Growing acquires or upgrades locks before any release occurs.

C - Incorrect: Freely acquiring after release violates the defining 2PL phase restriction.

D - Incorrect: Growing is precisely the phase in which lock requests may be made.

Primary note proof

Study-note section ADB12

> Basic 2PL has a growing phase that acquires/upgrades locks and a shrinking phase that releases/downgrades them. Once any lock is released, no new lock may be acquired. The lock point is the instant the final lock is obtained. 2PL guarantees conflict serializability for well-formed transactions, but can deadlock and basic 2PL alone need not prevent cascading aborts.

Study-note section ADB12

> Two-phase locking orders access to data; two-phase commit coordinates a distributed commit decision.

Study-note section ADB12

> Conservative/static 2PL obtains all needed locks before execution and does not begin while only a subset is available, preventing hold-and-wait deadlock.

Question sources

Advanced Database Systems review questions · Question 16

Question 200 | NS0122

Which concurrency control technique uses start-times to order transaction execution?

A. Locking

B. Optimistic Concurrency Control

C. Timestamp-based Concurrency Control

D. Shadow Paging

Hint

Compare the transaction timestamp with the item's recorded read and write timestamps.

Answer C: Timestamp-based concurrency control.

ADB15 assigns each transaction a timestamp and checks reads and writes against item read/write timestamps. These checks preserve timestamp order and can abort stale operations. Locking instead uses ownership compatibility, optimistic control validates private work and shadow paging concerns recovery page maps. Only C names the start-time ordering technique described.

Why each choice is right or wrong

A - Incorrect: Locking uses lock compatibility and protocols rather than being defined by transaction start-time ordering.

B - Incorrect: Optimistic control validates private read/write sets rather than applying these timestamp-order tests.

C - Correct: Timestamp-based control orders valid conflicting operations using transaction timestamps.

D - Incorrect: Shadow paging preserves committed page maps for recovery rather than ordering concurrent operations.

Primary note proof

Study-note section ADB15

> Assign each transaction TS(T). For item X track read_TS(X), the largest timestamp of a successful reader, and write_TS(X), that of its latest successful writer. A read by T aborts if TS(T)<write_TS(X); otherwise it succeeds and updates read_TS. A write aborts if TS(T)<read_TS(X) or TS(T)<write_TS(X); otherwise it succeeds and sets write_TS. These tests preserve timestamp order.

Study-note section ADB16

> Optimistic/validation control reads and computes privately, validates its read/write sets against overlapping transactions, then publishes if validation succeeds; otherwise it retries.

Study-note section ADB29

> Shadow/copy-on-write update writes a new location while preserving the committed version.

Related reading (context, not primary note proof)

Study-note section ADB11

Question sources

Advanced Database Systems review questions · Question 17

Question 201 | NS0123

Optimistic Concurrency Control (OCC) is best suited for:

A. High-conflict write-heavy workloads

B. Read-heavy workloads where conflicts are rare

C. Single-user database systems

D. Distributed transactions only

Hint

Consider conflicts, private work and validation costs, or lock-management overhead.

Answer B: Read-heavy, low-conflict workloads.

ADB16 computes privately, validates and publishes, retrying failed validation. Low-conflict workloads let most work succeed; high write contention wastes computation on aborts. B states that criterion explicitly. Single-user systems do not need intertransaction conflict control, and OCC is not restricted to distributed transactions. Actual conflict frequency matters more than the read-heavy label alone.

Why each choice is right or wrong

A - Incorrect: High conflicts cause validation failures and repeated wasted work.

B - Correct: Rare conflicts allow optimistic private work to validate successfully with few retries.

C - Incorrect: A single-user setting lacks the concurrent conflict problem motivating this choice.

D - Incorrect: Optimistic control is not limited to distributed transaction environments.

Primary note proof

Study-note section ADB16

> Optimistic/validation control reads and computes privately, validates its read/write sets against overlapping transactions, then publishes if validation succeeds; otherwise it retries. It works well when conflicts are uncommon, but high contention wastes work on aborts. Multiversion control keeps versions so a reader can use an appropriate snapshot. Snapshot isolation is not automatically serializable: concurrent writes to different rows can violate a shared invariant.

Question sources

Advanced Database Systems review questions · Question 18

Question 202 | NS0124

Deadlock occurs when:

A. One transaction waits for a resource held by no one

B. Two or more transactions each wait for resources held by the other in a circular chain

C. A transaction commits too slowly

D. Multiple transactions read identical data

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer B: A circular resource-wait chain.

ADB13 requires a waiting cycle: participants hold resources needed by others and cannot make progress without a victim release or another intervention. Waiting for an unheld resource does not state such a cycle. Slow commit and simultaneous reads can occur without deadlock; compatible shared readers in particular need not block one another.

Why each choice is right or wrong

A - Incorrect: A resource held by no one does not create the stated circular hold-and-wait dependency.

B - Correct: Each participant waits for another holder in a cycle, preventing independent progress.

C - Incorrect: A slow commit may still progress and need not form a waiting cycle.

D - Incorrect: Compatible shared reads of the same data do not inherently deadlock.

Primary note proof

Study-note section ADB13

> Deadlock requires a waiting cycle: T1 holds X and wants Y; T2 holds Y and wants X. In a wait-for graph, an edge Ti→Tj means Ti waits for a lock held by Tj. A cycle indicates deadlock under the ordinary lock-wait model. Abort a suitable victim, undo its effects and release its locks; repeat if other cycles remain. Aborting every deadlocked transaction also breaks cycles but wastes more work.

Study-note section ADB13

> A timeout aborts long waits but may mistake slow progress for deadlock.

Study-note section ADB11

> No double-locking is not a prohibition on shared readers.

Question sources

Advanced Database Systems review questions · Question 19

Question 203 | NS0125

In the Wait-Die protocol, when a YOUNGER transaction requests a resource held by an OLDER transaction:

A. The younger transaction dies (aborts)

B. The older transaction dies

C. The younger transaction waits

D. Both transactions abort

Hint

Identify requester and holder ages before applying the protocol's action.

Answer A: The younger requester aborts.

The requester is younger and the holder older. ADB14 makes that requester die in wait-die; only an older requester may wait for a younger holder. On retry the aborted transaction retains its original timestamp. Wounding a holder belongs to a different protocol, and both participants need not be aborted.

Why each choice is right or wrong

A - Correct: Wait-die aborts the younger requester when the needed resource is held by an older transaction.

B - Incorrect: The older holder keeps its resource; this protocol does not kill it for a younger requester.

C - Incorrect: Waiting is allowed for an older requester facing a younger holder, not this case.

D - Incorrect: The policy aborts the younger requester, not both participants.

Primary note proof

Study-note section ADB14

> A smaller timestamp means an older transaction. In wait-die, an older requester may wait for a younger holder; a younger requester needing an older holder aborts itself. In wound-wait, an older requester aborts the younger holder; a younger requester waits for the older holder. Always identify requester versus holder before applying the names.

Study-note section ADB14

> Retain the original timestamp when restarting an aborted transaction so it gains relative priority as new transactions arrive.

Study-note section ADB14

> In wait-die, an older transaction may wait for a younger one; a younger transaction requesting a resource held by an older one aborts.

Question sources

Advanced Database Systems review questions · Question 20

Question 204 | NS0126

In the Wound-Wait protocol, when an OLDER transaction requests a resource held by a YOUNGER transaction:

A. The older transaction waits

B. The younger is wounded (aborted), allowing the older to proceed

C. Both transactions are aborted

D. A new transaction takes over

Hint

Identify requester and holder ages before applying the protocol's action.

Answer B: Abort the younger holder so the older requester can proceed.

ADB14 says an older requester wounds a younger holder, while a younger requester waits. B is therefore the rule for the given ages. The protocol uses priority to prevent cycles rather than arbitrarily aborting both or handing the work to a new transaction. Any retry should preserve the original timestamp to reduce starvation.

Why each choice is right or wrong

A - Incorrect: An older requester wounds the younger holder rather than waiting under this policy.

B - Correct: The younger holder is aborted, releasing the resource for the older requester.

C - Incorrect: Aborting both is unnecessary and is not the wound-wait rule.

D - Incorrect: No replacement transaction takes over; the older requester continues after the conflicting holder releases.

Primary note proof

Study-note section ADB14

> A smaller timestamp means an older transaction. In wait-die, an older requester may wait for a younger holder; a younger requester needing an older holder aborts itself. In wound-wait, an older requester aborts the younger holder; a younger requester waits for the older holder. Always identify requester versus holder before applying the names.

Study-note section ADB14

> Retain the original timestamp when restarting an aborted transaction so it gains relative priority as new transactions arrive.

Study-note section ADB14

> In wait-die, an older transaction may wait for a younger one; a younger transaction requesting a resource held by an older one aborts.

Question sources

Advanced Database Systems review questions · Question 21

Question 205 | NS0127

Lock Timeout as a deadlock solution works by:

A. Building and analyzing wait-for graphs

B. Aborting any transaction that waits longer than a specified time limit

C. Locking all items before a transaction starts

D. Assigning age-based priorities

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer B: Abort a transaction whose wait exceeds the configured limit.

ADB13 defines timeout by excessive wait duration and warns it can mistake slow progress for deadlock. B matches that mechanism. Wait-for graphs detect cycles, conservative locking obtains all locks before execution and age priorities define wait-die/wound-wait decisions. Those are distinct policies rather than implementations of a time limit.

Why each choice is right or wrong

A - Incorrect: Graph analysis detects cycles rather than deciding from elapsed lock-wait time.

B - Correct: Timeout aborts a waiting transaction after its configured wait limit is exceeded.

C - Incorrect: Preacquiring all locks is conservative locking rather than timeout handling.

D - Incorrect: Age-based priorities define other prevention policies, not an elapsed-wait threshold.

Primary note proof

Study-note section ADB13

> Deadlock requires a waiting cycle: T1 holds X and wants Y; T2 holds Y and wants X. In a wait-for graph, an edge Ti→Tj means Ti waits for a lock held by Tj. A cycle indicates deadlock under the ordinary lock-wait model. Abort a suitable victim, undo its effects and release its locks; repeat if other cycles remain. Aborting every deadlocked transaction also breaks cycles but wastes more work.

Study-note section ADB13

> A timeout aborts long waits but may mistake slow progress for deadlock.

Study-note section ADB12

> Conservative/static 2PL obtains all needed locks before execution and does not begin while only a subset is available, preventing hold-and-wait deadlock.

Related reading (context, not primary note proof)

Study-note section ADB14

Question sources

Advanced Database Systems review questions · Question 22

Question 206 | NS0128

A schedule is serializable if:

A. All transactions execute one after another with no overlap

B. Its concurrent execution produces the same result as some serial execution of those transactions

C. All transactions commit at the same time

D. No two transactions access the same table

Hint

Distinguish transaction ordering, conflict conditions and full execution equivalence.

Answer B: Equivalent behavior to some serial execution, even if interleaved.

ADB09 defines a serializable schedule as equivalent to a serial order while permitting interleaving. B matches when same result means the required full transaction equivalence, not an accidental equality of one final value. A describes a serial schedule, which is a special case but not the general definition. Commit simultaneity and disjoint tables are not required.

Why each choice is right or wrong

A - Incorrect: This defines serial execution, a stricter special case that omits permitted serializable interleavings.

B - Correct: Serializable behavior is equivalent to some serial ordering; coincidental equality of one value alone is insufficient.

C - Incorrect: Simultaneous commits do not establish the required transaction-execution equivalence.

D - Incorrect: Serializable transactions may access the same tables if their operations are correctly ordered.

Primary note proof

Study-note section ADB09

> A serial schedule executes each transaction's operations consecutively. A serializable schedule is equivalent to some serial order; it may interleave transactions. Equality of one final value is insufficient evidence because different executions can accidentally coincide. Two operations conflict when they belong to different transactions, access the same item and at least one writes. Read/read pairs do not conflict.

Question sources

Advanced Database Systems review questions · Question 23

Question 207 | NS0129

Lock Upgrade refers to:

A. Releasing an exclusive lock to a shared lock

B. Converting a shared (read) lock to an exclusive (write) lock

C. Removing all locks at commit

D. Granting a lock to multiple transactions

Hint

Compare access permissions and same-item compatibility before granting a lock request.

Answer B: Convert Shared to Exclusive.

ADB11 identifies S→X as an upgrade, possible only after incompatible other holders leave. It strengthens an existing read lock to permit writing. X→S is a downgrade, releasing all locks is transaction cleanup and granting compatible readers is shared acquisition. Only B names the requested conversion of a held lock.

Why each choice is right or wrong

A - Incorrect: Exclusive-to-shared is a downgrade rather than an upgrade.

B - Correct: Shared-to-exclusive strengthens the lock to permit writing when compatibility allows.

C - Incorrect: Removing locks is release rather than conversion to a stronger mode.

D - Incorrect: Granting compatible shared readers is not the upgrade of one existing lock.

Primary note proof

Study-note section ADB11

> A transaction must own the appropriate lock before access and can release only its own locks. S→X is an upgrade, possible only after conflicting other holders leave. X→S is a downgrade. The lock manager decides requests; the lock table stores granted locks and waiting requests. One transaction may request a lock on an already locked item: it is granted if compatible, otherwise waits or aborts under the policy. No double-locking is not a prohibition on shared readers.

Question sources

Advanced Database Systems review questions · Question 24

Question 208 | NS0130

Starvation in concurrency control means:

A. A transaction holding too many locks

B. A transaction is perpetually delayed and never executes due to other transactions always taking priority

C. The database runs out of storage space

D. A transaction commits before completing all operations

Hint

Distinguish a waiting cycle, a long wait and indefinite unfair postponement.

Answer B: Indefinite postponement caused by repeated priority to other work.

ADB13 defines starvation as indefinite postponement while other transactions progress, without requiring a deadlock cycle. B states that repeated bypass precisely. Holding many locks can create contention but does not itself prove starvation. Storage exhaustion and premature commit are different failures, so neither supplies the scheduling-fairness definition.

Why each choice is right or wrong

A - Incorrect: Holding many locks describes resource use rather than being perpetually denied execution.

B - Correct: Repeated priority to others can indefinitely postpone the transaction while other work continues.

C - Incorrect: Running out of storage is a capacity failure rather than the defined scheduling starvation.

D - Incorrect: Premature commit concerns transaction correctness rather than indefinite postponement.

Primary note proof

Study-note section ADB13

> Fixed resource ordering and conservative 2PL prevent cycles. A no-wait policy aborts immediately on an incompatible lock request and needs no age timestamp. A timeout aborts long waits but may mistake slow progress for deadlock. Starvation is indefinite postponement while other work proceeds, without a required cycle. Fair queues, sensible victim selection and retaining original transaction timestamps on retry reduce starvation; deadlock freedom alone does not prove fairness.

Question sources

Advanced Database Systems review questions · Question 25

Question 209 | NS0131

The first step in SQL query processing is:

A. Optimization

B. Evaluation

C. Parsing and Translation

D. Result Delivery

Hint

Separate query representation, strategy selection and execution.

Answer C: Parsing and translation.

ADB17 first checks syntax and semantics and constructs a logical query representation, before optimization and evaluation. C groups those initial activities. Result delivery follows execution, while optimization needs an understood query to compare valid plans. The phase grouping is conventional, but the ordering dependency is explicit in the complete note.

Why each choice is right or wrong

A - Incorrect: Optimization follows understanding and representing the query rather than preceding it.

B - Incorrect: Evaluation executes a selected plan and is not the first processing activity.

C - Correct: Parsing/translation establishes query meaning and logical representation before optimization.

D - Incorrect: Results can be delivered only after the required evaluation produces them.

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB17

> Query processing is the whole activity; selecting a suitable efficient strategy is optimization.

Study-note section ADB17

> DDL is not a mandatory stage before each SELECT, so authentication→DDL→DML permutations do not describe the processing sequence.

Question sources

Advanced Database Systems review questions · Question 26

Question 210 | NS0132

SQL queries are internally translated into:

A. Machine code

B. Relational algebra

C. Python scripts

D. XML documents

Hint

Separate query representation, strategy selection and execution.

Answer B: Relational algebra.

ADB17 translates a query into relational algebra or a logical query tree for equivalent rewrites and plan selection. B names that logical representation. A particular engine may later compile execution code, but machine code, Python scripts and XML documents are not the general logical translation target taught here. Logical meaning precedes implementation-specific execution choices.

Why each choice is right or wrong

A - Incorrect: Machine code may be an implementation detail but is not the stated logical translation target.

B - Correct: Relational algebra represents the logical operations used for rewriting and planning.

C - Incorrect: Python scripts are not the general internal representation specified here.

D - Incorrect: XML documents are not the defined general relational query representation.

Primary note proof

Study-note section ADB17

> Parsing checks SQL grammar; semantic analysis resolves names, types and access permissions. Translation constructs a logical representation, typically relational algebra or a logical query tree. Optimization chooses equivalent operations, join order and physical access methods. Evaluation executes that plan and returns results. Not every implementation exposes exactly the same phase boundaries.

Study-note section ADB17

> A physical execution plan adds scan/index choices and algorithms.

Study-note section ADB17

> A query tree has base relations at leaves and operators at internal nodes; a query graph emphasizes relations and predicates.

Question sources

Advanced Database Systems review questions · Question 27

Question 211 | NS0133

What does query optimization aim to minimize?

A. The number of tables referenced

B. The overall cost (time and resources) of executing the query

C. The number of join operations

D. The character length of the SQL query

Hint

Identify the resource objective, counts and unit costs before comparing execution strategies.

Answer B: Overall execution cost under the chosen resource model.

ADB19 measures candidate plans using I/O, CPU, memory, temporary storage and network traffic. The aim is a suitable low-cost equivalent plan, not syntactic brevity or indiscriminate removal of joins and tables. Required tables and semantics must be preserved; even fewer operations do not necessarily imply lower runtime or resource cost.

Why each choice is right or wrong

A - Incorrect: Required tables cannot be discarded merely to reduce the table count.

B - Correct: Optimization compares equivalent plans using estimated time and resource costs.

C - Incorrect: Join count alone ignores access paths, cardinalities and the cost of each join.

D - Incorrect: Short SQL text does not necessarily produce a cheaper execution plan.

Primary note proof

Study-note section ADB19

> A cost model estimates resources for a candidate plan: page I/O, CPU, memory, temporary storage and network traffic. Catalog statistics include row/page counts, distinct values and distributions. Selection with fraction s predicts s×N rows. For non-NULL equijoin keys, a common estimate is $|R|\times|S|/\max(V(R,k),V(S,k))$, assuming uniform values and containment of the smaller distinct-value domain within the larger. Adjust counts for NULLs; if either non-NULL input is empty, the result is zero. Skew, correlations or incomplete domain overlap can invalidate this estimate; known disjoint domains give zero matches.

Study-note section ADB19

> Disk I/O often dominates textbook local-disk examples; CPU can dominate in-memory work and network can dominate distributed plans.

Study-note section ADB19

> For a simple rotating-disk model, cost = seeks×seek_time + transferred_blocks×block_transfer_time.

Question sources

Advanced Database Systems review questions · Question 28

Question 212 | NS0134

The dominant cost factor in query execution is typically:

A. CPU processing time

B. Network transmission time

C. Disk I/O time (seek + transfer)

D. Memory allocation time

Hint

Identify the resource objective, counts and unit costs before comparing execution strategies.

Answer C: Disk I/O in the conventional local-disk query-cost model.

ADB19 says disk I/O often dominates textbook local-disk examples, matching the question's typical teaching model. C includes both seek and transfer work. This is a contextual tendency, not a universal claim: CPU can dominate in-memory work and communication can dominate distributed plans. The answer should retain that local-disk qualification rather than extend it to every system.

Why each choice is right or wrong

A - Incorrect: CPU can dominate in-memory work but is not the typical local-disk-model answer.

B - Incorrect: Network cost can dominate distributed plans rather than the conventional local-disk model.

C - Correct: Seek and transfer I/O often dominate the conventional disk-based cost model.

D - Incorrect: Memory allocation is not the dominant term specified for the textbook local-disk model.

Primary note proof

Study-note section ADB19

> For a simple rotating-disk model, cost = seeks×seek_time + transferred_blocks×block_transfer_time. Five seeks at 5 ms plus ten blocks at 1 ms cost 35 ms. Disk I/O often dominates textbook local-disk examples; CPU can dominate in-memory work and network can dominate distributed plans. State the assumed cost objective before ranking strategies.

Question sources

Advanced Database Systems review questions · Question 29

Question 213 | NS0135

If a query requires 5 seeks (tS=5ms each) and transfers 10 blocks (tT=1ms each), what is the disk I/O cost?

A. 50ms

B. 35ms

C. 15ms

D. 25ms

Hint

Identify the resource objective, counts and unit costs before comparing execution strategies.

Answer B: 35 ms.

ADB19 adds seeks times seek duration to blocks times per-block transfer duration. Five seeks cost 5×5=25 ms and ten block transfers cost 10×1=10 ms. Their sum is 35 ms. The terms must be added, not multiplied together or considered separately; this is the supplied simple disk model rather than a measured universal query runtime.

Why each choice is right or wrong

A - Incorrect: Fifty does not equal the sum of 25 ms seeking and 10 ms transfer.

B - Correct: Five times five plus ten times one gives 35 ms.

C - Incorrect: Fifteen omits the correct five-seek multiplication.

D - Incorrect: Twenty-five counts only seek time and omits ten milliseconds of transfer.

Primary note proof

Study-note section ADB19

> For a simple rotating-disk model, cost = seeks×seek_time + transferred_blocks×block_transfer_time. Five seeks at 5 ms plus ten blocks at 1 ms cost 35 ms. Disk I/O often dominates textbook local-disk examples; CPU can dominate in-memory work and network can dominate distributed plans. State the assumed cost objective before ranking strategies.

Question sources

Advanced Database Systems review questions · Question 30

Question 214 | NS0136

Database recovery restores the database after failure to:

A. Its state at database creation

B. A consistent and correct state, undoing incomplete transactions and reapplying committed ones

C. The state at the last user login

D. An empty state for fresh start

Hint

Identify what failed, what durable resources survive and which restoration action is required.

Answer B: A valid recovered state preserving durable commits and removing incomplete effects.

ADB26 defines restart recovery by preserving durable commits and removing incomplete effects after failure. B matches the simple undo/redo teaching model. The detailed work depends on buffer policy and algorithm; not every committed update needs physical redo if already installed. Recovery does not reset the database to creation, last login or an empty replacement state.

Why each choice is right or wrong

A - Incorrect: Returning to creation would discard legitimate later durable commits.

B - Correct: Recovery preserves committed work and removes incomplete effects to restore valid state.

C - Incorrect: User login time is not the transaction recovery boundary.

D - Incorrect: Emptying the database would destroy committed data rather than recover it.

Primary note proof

Study-note section ADB26

> Transaction failures include invalid operations, constraint violations and concurrency-control aborts. Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure. A system crash loses volatile memory while disks may survive. Media failure damages persistent storage and normally requires restoring a backup, then replaying available archived logs. Human mistakes may require point-in-time restoration before the bad committed operation.

Study-note section ADB26

> Restart recovery restores a valid database after a crash while preserving durable commits and removing incomplete effects.

Study-note section ADB27

> In a simple undo/redo teaching model, undo losers and redo winners.

Question sources

Advanced Database Systems review questions · Question 31

Question 215 | NS0137

Which type of failure occurs when DBMS software bugs cause unexpected behavior?

A. Hardware Failure

B. Natural Disaster

C. Software Error

D. Network Failure

Hint

Classify the stated cause rather than inferring it from the observed symptom.

Answer C: Software error.

The stem directly identifies DBMS software bugs as the cause. That is a software error rather than failed hardware, a natural event or broken communication. ADB26 separates transaction/system/media failure concerns so the recovery resource can be chosen appropriately; an observed crash alone would not identify its cause, but this question explicitly supplies it.

Why each choice is right or wrong

A - Incorrect: The stated cause is faulty software behavior rather than a malfunctioning hardware component.

B - Incorrect: No natural event is described; the prompt names a software bug.

C - Correct: Unexpected behavior caused by DBMS bugs is a software error.

D - Incorrect: No failed network connection is given as the cause of the behavior.

Primary note proof

Study-note section ADB-S05

> A software failure originates in erroneous program behavior, such as a logical programming defect, invalid operation or operating-system fault. A hardware failure originates in malfunctioning physical equipment; loss of disk contents is media failure. An accidental user deletion is human error even if the software executes the requested command correctly. Physical catastrophes such as fire can also cause failure. The observed symptom of a crash does not alone identify which cause occurred.

Study-note section ADB26

> Generic recovery-manager terminology should not be mistaken for an exact MySQL component specification.

Study-note section ADB26

Enforcing concurrency can deliberately abort work; it does not prevent every transaction failure.

Question sources

Advanced Database Systems review questions · Question 32

Question 216 | NS0138

What does WAL require? In B, 'corresponding data changes' specifically means durable output of the changed data page; volatile buffer changes may occur earlier.

A. Data written to disk before the log record

B. Log records written to stable storage BEFORE corresponding data changes

C. Logs written only after transaction commits

D. Checkpoints before every write operation

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer B: B. Log records written to stable storage BEFORE corresponding data changes

The update log needed for recovery must be stable before the changed data page becomes persistent. B states that order under the question's explicit durable-write meaning. The buffer can be modified earlier. Data-before-log reverses the rule, waiting until after commit is not the required logging policy and checkpointing before every write is not a WAL requirement.

Why each choice is right or wrong

A - Incorrect: The durable log must precede the changed page, not follow it.

B - Correct: With the stated scope, stable logs precede corresponding durable page output.

C - Incorrect: Update logging is not deferred until after commit.

D - Incorrect: WAL does not require a checkpoint for every write.

Primary note proof

Study-note section ADB27

> Write-ahead logging requires the update log needed for recovery to reach stable storage before the corresponding changed data page reaches disk. To acknowledge a durable commit, force the transaction's required log records, including its commit record, to stable storage. A buffer can be changed before that log flush; WAL orders durable writes, not every CPU assignment.

Study-note section ADB27

> Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not.

Study-note section ADB27

> Redo installs logged after-images when necessary, usually forward.

Question sources

Advanced Database Systems review questions · Question 33

Question 217 | NS0139

The Redo operation during recovery:

A. Reverses incomplete transactions

B. Re-applies committed transactions whose changes were not yet persisted to disk

C. Creates new checkpoints

D. Locks data items during the recovery process

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer B: Reapply missing logged committed effects in the simple undo/redo model.

ADB27 uses redo to install logged after-images when necessary. B describes preserving committed work missing from disk pages in the simple teaching model. It does not mean re-running application side effects or blindly adding an amount twice. ARIES has a broader repeat-history phase before undo, but that does not make reversal, checkpoint creation or lock acquisition the definition of redo.

Why each choice is right or wrong

A - Incorrect: Reversing incomplete changes is undo rather than redo.

B - Correct: Redo installs required logged effects missing from data pages, preserving committed work in the simple model.

C - Incorrect: Checkpoint creation records recovery state rather than reapplying update after-images.

D - Incorrect: Locking may coordinate access but is not the operation of logged redo.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB30

> In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo.

Study-note section ADB30

> Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Related reading (context, not primary note proof)

Study-note section ADB11

Question sources

Advanced Database Systems review questions · Question 34

Question 218 | NS0140

The Undo operation during recovery:

A. Re-applies committed transactions

B. Reverses changes of incomplete/aborted transactions using before-images from the log

C. Writes the WAL to stable storage

D. Creates shadow page copies

Hint

Track durable-write ordering and distinguish reversing incomplete work from installing logged effects.

Answer B: Restore before-images of incomplete or aborted work.

ADB27 restores before-images when undoing incomplete effects, normally following that transaction's updates backward. B states both the affected work and the supporting log information. Applying committed after-images is redo, forcing logs is a WAL durability action, and copying pages is part of a different update/recovery approach. None is the reversal operation itself.

Why each choice is right or wrong

A - Incorrect: Reapplying committed effects is the redo purpose in the simple model.

B - Correct: Undo restores logged old values to remove incomplete or aborted transaction effects.

C - Incorrect: Forcing the log supports durable ordering but does not itself reverse data changes.

D - Incorrect: Shadow-page copies preserve versions instead of defining logged before-image undo.

Primary note proof

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward. Redo installs logged after-images when necessary, usually forward. Setting X to its logged after-value is idempotent; blindly replaying X=X+10 twice is not. In a simple undo/redo teaching model, undo losers and redo winners. ARIES has a different redo phase that repeats history before undoing losers.

Study-note section ADB29

It is an alternative to conventional log-based recovery, with allocation, fragmentation and concurrency costs; it is not primarily a special action done only during recovery.

Study-note section ADB29

Shadow/copy-on-write update writes a new location while preserving the committed version.

Question sources

Advanced Database Systems review questions · Question 35

Question 219 | NS0141

Shadow Paging recovery technique works by:

A. Using timestamps to sequence recovery

B. Writing changes to shadow (copy) pages, keeping originals intact, then swapping on commit

C. Using transaction logs for all rollbacks

D. Taking checkpoints every transaction

Hint

Separate update timing from update location and follow the commit boundary.

Answer B: Copy changed pages and atomically install the new durable map on commit.

ADB29 keeps the committed page map unchanged while modifications use new locations and a modified map. Commit durably installs the new root/map pointer after required pages are durable. B captures that preservation-and-switch mechanism. Timestamp ordering, mandatory per-transaction checkpoints and universal log-based rollback are not the defining shadow-paging steps.

Why each choice is right or wrong

A - Incorrect: Timestamp ordering controls concurrent operations rather than defining shadow page-map replacement.

B - Correct: Copy-on-write preserves originals and commit safely installs the new page map.

C - Incorrect: Shadow paging is an alternative to conventional log-based recovery, not mandatory logged rollback for all changes.

D - Incorrect: A checkpoint every transaction is not the mechanism that preserves the old page map.

Primary note proof

Study-note section ADB29

In-place update overwrites the original location. Shadow/copy-on-write update writes a new location while preserving the committed version. Shadow paging keeps an unchanged committed page map and a modified map; commit durably installs a new root/map pointer atomically after required pages are durable. Before commit, a crash uses the old map. It is an

> alternative to conventional log-based recovery, with allocation, fragmentation and concurrency costs; it is not primarily a special action done only during recovery.

Study-note section ADB30

> Do not assume every transaction before a checkpoint needs no processing.

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history.

Related reading (context, not primary note proof)

Study-note section ADB15

Question sources

Advanced Database Systems review questions · Question 36

Question 220 | NS0142

A checkpoint in database recovery helps because:

A. It stops all transactions

B. Recovery can start from the checkpoint state rather than processing the entire log from the beginning

C. It creates a full database backup copy

D. It grants exclusive locks to all active transactions

Hint

Identify which history a recovery boundary can safely exclude and whether active work spans it.

Answer B: A checkpoint supplies recovery state that avoids scanning all historical logs from the beginning.

ADB30 uses durable checkpoint state to avoid processing all old history. B describes that benefit if start means entering recovery analysis, not assuming every earlier update is irrelevant. Active transactions can span a checkpoint and fuzzy-checkpoint redo may require earlier log positions. A checkpoint is not a full backup, universal stop or lock grant.

Why each choice is right or wrong

A - Incorrect: Fuzzy checkpoints permit ongoing transactions, so stopping all work is not required.

B - Correct: Recorded recovery state provides a bounded starting point, while earlier necessary undo/redo must still be considered.

C - Incorrect: A checkpoint stores recovery metadata and is not a full separate database backup.

D - Incorrect: Checkpointing does not grant every active transaction exclusive locks.

Primary note proof

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history. In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo. Active transactions can span the checkpoint, so their relevant earlier updates can still need undo. Do not assume every transaction before a checkpoint needs no processing.

Study-note section ADB30

> Recovery may start analysis at the checkpoint but redo from an earlier required log position.

Study-note section ADB30

> A checkpoint is neither a full backup nor an automatic commit of active transactions.

Related reading (context, not primary note proof)

Study-note section ADB11

Question sources

Advanced Database Systems review questions · Question 37

Question 221 | NS0143

Under conventional deferred update with no-force, which recovery classifications describe the policy? Select all that apply. Interpret NO UNDO / REDO as 'no undo, redo required'.

A. UNDO/REDO

B. UNDO only

C. REDO only

D. NO UNDO / REDO

Hint

Separate update timing from update location and follow the commit boundary.

Answer C, D: C. REDO only; D. NO UNDO / REDO

Because uncommitted changes have not reached the persistent database, crash recovery does not need to undo them there. No-force permits committed changes to remain uninstalled, so redo may be required. REDO only and NO UNDO / REDO express the same classification under the stated notation. Undo-only and undo/redo both assert an undo requirement that this deferred model avoids.

Why each choice is right or wrong

A - Incorrect: Usual deferred update does not require undo of uncommitted persistent database changes.

B - Incorrect: Undo only gives the opposite recovery requirement from the usual deferred no-force model.

C - Correct: REDO only is the usual deferred-update recovery classification.

D - Correct: NO UNDO/REDO is another spelling of the same redo-only requirement.

Primary note proof

Study-note section ADB29

> Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO. Redo-only and NO-UNDO/REDO mean the same thing, so those cannot be separate unique answers. Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Study-note section ADB28

> Force writes a transaction's updated data pages at commit; no-force permits committed dirty pages to remain buffered, hence recovery may need redo.

Study-note section ADB28

> Steal permits a page with uncommitted changes to be written to disk; hence recovery may need undo.

Question sources

Advanced Database Systems review questions · Question 38

Question 222 | NS0144

Which data update strategy writes changes to a SEPARATE disk location preserving the original?

A. Immediate Update

B. Deferred Update

C. Shadow Update

D. In-Place Update

Hint

Separate update timing from update location and follow the commit boundary.

Answer C: Shadow update, or copy-on-write.

ADB29 distinguishes where an update is written from when it may become persistent. Shadow/copy-on-write uses a new location while retaining the committed version, matching C. Immediate and deferred classify commit timing, not necessarily a separate location. In-place update overwrites the original location, which directly contradicts the preservation condition in the stem.

Why each choice is right or wrong

A - Incorrect: Immediate update concerns allowing persistent changes before commit rather than requiring new locations.

B - Incorrect: Deferred update concerns withholding uncommitted persistent effects rather than necessarily copying pages.

C - Correct: Shadow/copy-on-write preserves the original and writes the changed value at a separate location.

D - Incorrect: In-place update overwrites the original location rather than preserving it intact.

Primary note proof

Study-note section ADB29

> In-place update overwrites the original location. Shadow/copy-on-write update writes a new location while preserving the committed version. Shadow paging keeps an unchanged committed page map and a modified map; commit durably installs a new root/map pointer atomically after required pages are durable. Before commit, a crash uses the old map. It is an alternative to conventional log-based recovery, with allocation, fragmentation and concurrency costs; it is not primarily a special action done only during recovery.

Study-note section ADB29

> Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Study-note section ADB29

> Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO.

Question sources

Advanced Database Systems review questions · Question 39

Question 223 | NS0145

In the course's conceptual transaction-log model supporting undo of incomplete changes and redo of committed changes, what value information does a write-operation record store?

A. Only new (after-image) values

B. Only old (before-image) values

C. Both old (before-image) AND new (after-image) values

D. Only transaction IDs and timestamps

Hint

Trace movement between program variables, buffers and disk, or the meaning of a recorded transaction event.

Answer C: C. Both old (before-image) AND new (after-image) values

The given UPDATE record contains both the old and new values. The old value supplies a before-image for undo, while the new value supplies an after-image for redo. Therefore the record stores both kinds of image. Real systems can use other physical or logical formats, but that variation does not alter the explicit conceptual record being examined here.

Why each choice is right or wrong

A - Incorrect: The given format includes old as well as new, so after-images alone are incomplete.

B - Incorrect: The format includes new as well as old, so before-images alone are incomplete.

C - Correct: Both before-image and after-image fields are present.

D - Incorrect: Transaction identifiers alone omit the explicitly supplied value fields.

Primary note proof

Study-note section ADB07

> A conceptual log can contain START, UPDATE(T,X,old,new), COMMIT, ABORT and END records. Before-images support undo; after-images support redo. Real systems may use physical or logical records, and need not log every read. COMMIT records successful completion; an ABORT record can mark the start of rollback rather than prove every undo step is complete.

Study-note section ADB27

> Undo restores before-images of incomplete effects, usually traversing that transaction's updates backward.

Study-note section ADB27

> In a simple undo/redo teaching model, undo losers and redo winners.

Question sources

Advanced Database Systems review questions · Question 40

Question 224 | NS0146

A distributed database is best described as:

A. A single database on one powerful server

B. Multiple logically related databases across networked computers, managed transparently

C. A fully replicated database at every network location

D. A web-based file storage system

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer B: Logically related data across networked sites under coordinated database management.

ADB33 defines logically related data across networked sites managed as a coordinated system. B matches that architecture and its logical abstraction. Full replication is only one placement choice, not a defining requirement. A single powerful server is centralized, and generic web file storage does not establish coordinated database semantics across related distributed data.

Why each choice is right or wrong

A - Incorrect: A database confined to one server is centralized rather than distributed across networked sites.

B - Correct: Networked, logically related data with coordinated management fits the distributed database definition.

C - Incorrect: A distributed database need not replicate every row at every network location.

D - Incorrect: Generic web file storage does not by itself establish coordinated distributed database management.

Primary note proof

Study-note section ADB33

> A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB37

> Full replication does not inherently mean every copy is updated simultaneously; its synchronization rule must be stated.

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once.

Question sources

Advanced Database Systems review questions · Question 41

Question 225 | NS0147

A homogeneous distributed database system requires:

A. Different DBMS at each site

B. Same DBMS software at all sites

C. No network between sites

D. Full independence of each site

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer B: The same compatible DBMS technology at the sites.

ADB33 defines homogeneity through compatible DBMS technology across sites. B is the listed form of that condition. It does not imply identical local schemas, replication or immediate synchronization. Different DBMSs describe heterogeneity, while removing the network or demanding full independence contradicts coordinated distributed management rather than defining homogeneity.

Why each choice is right or wrong

A - Incorrect: Different DBMS technologies characterize heterogeneous integration rather than homogeneity.

B - Correct: The same DBMS software supplies the compatible database technology described by homogeneous systems.

C - Incorrect: Distributed sites need network communication for coordinated management.

D - Incorrect: Full local independence is an autonomy choice rather than a homogeneity requirement.

Primary note proof

Study-note section ADB33

> A distributed database contains logically related data across networked sites, managed as a coordinated system. A homogeneous system uses compatible DBMS technology across sites; a heterogeneous system integrates differing DBMSs, data models, schemas or interfaces. Different operating systems are possible but do not define the distinction. Same DBMS does not imply identical local schemas, automatic replication or immediate synchronization.

Study-note section ADB34

> These describe communication responsibilities, not fragmentation types.

Study-note section ADB34

State how much schema, transaction and administrative autonomy local sites retain.

Question sources

Advanced Database Systems review questions · Question 42

Question 226 | NS0148

Heterogeneous distributed databases use which standards for inter-database communication?

A. Same DBMS software only

B. Direct SQL between databases

C. ODBC and JDBC generic connectivity

D. LAN protocols exclusively

Hint

Separate coordinated network placement from engine compatibility and semantic integration.

Answer C: ODBC and JDBC connectivity interfaces.

ADB33 explicitly names ODBC and JDBC as connectivity interfaces for differing systems. C therefore identifies the offered communication interfaces. They do not automatically reconcile schemas, units or transaction semantics and do not create a distributed transaction by themselves. Identical DBMS software is not heterogeneity, and generic LAN protocols do not replace database-level connectivity interfaces.

Why each choice is right or wrong

A - Incorrect: Requiring the same DBMS removes the differing-engine situation rather than identifying its connectivity interfaces.

B - Incorrect: Direct SQL alone does not identify the generic cross-system connectivity standards named in the note.

C - Correct: ODBC and JDBC provide generic database connectivity while semantic mappings remain necessary.

D - Incorrect: LAN protocols alone do not supply the database API/interface compatibility described.

Primary note proof

Study-note section ADB33

> Heterogeneous design must reconcile naming, types, units, identifiers, query capabilities and transaction semantics. ODBC and JDBC provide connectivity interfaces; they do not automatically solve semantic differences or create one distributed transaction. Locality, expansion and replicated availability are potential benefits. Distribution adds communication, coordination and security costs; a site outage can still make needed data unavailable if no usable replica exists.

Related reading (context, not primary note proof)

Study-note section ADB24

Question sources

Advanced Database Systems review questions · Question 43

Question 227 | NS0149

Horizontal fragmentation divides a database table by:

A. Columns (attributes)

B. Rows based on a condition or criterion

C. Both rows and columns simultaneously

D. Database schemas

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer B: Rows selected by conditions.

ADB36 defines horizontal fragmentation by selecting row subsets with predicates. B directly states that operation. Column division is vertical fragmentation, and combining both dimensions is hybrid fragmentation. Schema names and geographic placement can influence a design, but are not the definition of what is split by horizontal fragmentation.

Why each choice is right or wrong

A - Incorrect: Splitting attributes is vertical rather than horizontal fragmentation.

B - Correct: Horizontal fragmentation partitions rows according to stated predicates or criteria.

C - Incorrect: Splitting both dimensions combines horizontal and vertical operations in hybrid fragmentation.

D - Incorrect: Schemas are not the row units divided by the defined horizontal operation.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB36

> Hybrid fragmentation combines row and column splits in either suitable order.

Study-note section ADB36

> Reconstruct by reversing the operations: key joins for corresponding vertical pieces and unions for horizontal pieces.

Question sources

Advanced Database Systems review questions · Question 44

Question 228 | NS0150

To reconstruct a horizontally fragmented table, which operation is used?

A. JOIN

B. UNION

C. INTERSECTION

D. DIFFERENCE

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer B: UNION.

Horizontal fragments contain selected subsets of the original relation's rows. ADB36 reconstructs the relation by their union, assuming completeness and the intended overlap rules. Joining is used to align vertical column pieces by a key; intersection would keep only shared rows and difference would remove rows, so neither reconstructs all horizontal fragments.

Why each choice is right or wrong

A - Incorrect: Key joins reconstruct matching vertical column pieces, not ordinary horizontal row partitions.

B - Correct: Union collects the rows from every complete horizontal fragment.

C - Incorrect: Intersection keeps only common rows and would discard rows unique to individual fragments.

D - Incorrect: Difference removes rows instead of recombining all fragment members.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB36

> Reconstruct by reversing the operations: key joins for corresponding vertical pieces and unions for horizontal pieces.

Study-note section ADB-S01

> A relation is represented by a table of tuples.

Question sources

Advanced Database Systems review questions · Question 45

Question 229 | NS0151

Vertical fragmentation divides a table by:

A. Rows

B. Columns (attributes), with primary key repeated in each fragment

C. Number of concurrent transactions

D. Geographic location of data

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer B: Column subsets with a repeated key or row identifier.

ADB36 splits a relation by columns and repeats a key or row identifier so the pieces can be joined without spurious matches. B describes that usual primary-key form. Rows are split horizontally; transaction counts and geography concern workload or placement rather than the attribute-level split defining vertical fragmentation. The shared identifier can be an appropriate equivalent key.

Why each choice is right or wrong

A - Incorrect: Row subsets define horizontal rather than vertical fragmentation.

B - Correct: Vertical fragments contain column subsets plus a repeated identifier for reconstruction.

C - Incorrect: Concurrent transaction count is a workload property rather than the dimension being split.

D - Incorrect: Geographic placement does not define the column projection used for vertical fragments.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB36

> Hybrid fragmentation combines row and column splits in either suitable order.

Study-note section ADB36

> Reconstruct by reversing the operations: key joins for corresponding vertical pieces and unions for horizontal pieces.

Question sources

Advanced Database Systems review questions · Question 46

Question 230 | NS0152

Full replication in distributed databases has which disadvantage?

A. Low data availability

B. Every update must be applied to all copies, creating high overhead

C. Fast read performance

D. Simple concurrency control

Hint

Separate copy placement, propagation timing and update-maintenance cost.

Answer B: Propagating each update across all copies adds maintenance overhead.

ADB37 says copies improve locality and can improve availability but require storage and update propagation. B names that maintenance burden for full replication. Propagation timing may be synchronous or asynchronous, so the answer does not imply every copy is updated simultaneously. Fast reads are a possible benefit, and concurrency control becomes more involved rather than inherently simple.

Why each choice is right or wrong

A - Incorrect: Usable replicas can improve availability rather than inherently making it low.

B - Correct: All copies must eventually reflect relevant updates, creating propagation and coordination overhead.

C - Incorrect: Fast reads are a potential benefit rather than the listed disadvantage.

D - Incorrect: Maintaining consistent copies does not inherently simplify concurrency control.

Primary note proof

Study-note section ADB37

> Full replication places the whole database at every relevant site; partial replication copies selected fragments to selected sites; nonredundant allocation stores each fragment once. Copies improve read locality and can improve availability but require storage and update propagation. They can also spread mistakes, so replication does not replace independent backups.

Study-note section ADB37

> Full replication does not inherently mean every copy is updated simultaneously; its synchronization rule must be stated.

Study-note section ADB37

> Lazy/asynchronous replication propagates later and may expose stale reads or need conflict resolution.

Question sources

Advanced Database Systems review questions · Question 47

Question 231 | NS0153

Fragmentation transparency in distributed databases means:

A. Users can see all the fragmentation details

B. Users interact with the database as a single entity, unaware of fragmentation existence

C. Only DBAs know about fragments

D. Fragments are listed as separate tables in the schema

Hint

Identify exactly which detail is hidden from the user.

Answer B: Users query the logical relation without knowing its fragments.

ADB35 defines fragmentation transparency as hiding the existence and reconstruction of fragments. B states that user-facing abstraction directly. It does not require literally only DBAs to know the implementation; developers and administrators may also need that knowledge. Exposing fragment tables or all fragmentation details to users would make fragmentation visible instead of transparent.

Why each choice is right or wrong

A - Incorrect: Showing all fragmentation details is the opposite of hiding them from ordinary queries.

B - Correct: Users operate on the whole logical relation without managing its fragment reconstruction.

C - Incorrect: Transparency describes the user abstraction, not exclusive knowledge restricted to DBAs alone.

D - Incorrect: Naming fragments separately exposes fragmentation rather than concealing it from the querying user.

Primary note proof

Study-note section ADB35

> Location transparency hides which site holds a named object. Fragmentation transparency hides the existence and reconstruction of fragments: users query the whole logical relation. Replication transparency hides which copy is used and how copies are maintained. The broad single-logical-database experience may involve all three.

Study-note section ADB35

> If users explicitly name a fragment but need not specify its site, fragmentation is visible while location can remain transparent.

Study-note section ADB35

> Hiding this column split from queries is fragmentation transparency; the split itself is a data-design operation.

Question sources

Advanced Database Systems review questions · Question 48

Question 232 | NS0154

The Middleware architecture in distributed DBMS uses:

A. Direct client-to-server connections

B. All servers collaborating on every query

C. A dedicated middleware server that mediates between clients and multiple independent database servers

D. A single centralized database server

Hint

Identify which tier submits, mediates or executes a request.

Answer C: Middleware mediates between clients and independent database servers.

ADB34 lets middleware translate requests, locate data, coordinate operations and combine results across independent servers. C describes that mediating tier. The logical responsibility does not require exactly one physical machine in every implementation. Direct connections omit this mediator, universal server collaboration is a different arrangement, and one centralized server is not the stated distributed middleware design.

Why each choice is right or wrong

A - Incorrect: Direct client-server links alone omit the middleware mediation that defines the requested architecture.

B - Incorrect: Every server need not participate in every query; relevant sources depend on the request.

C - Correct: A middleware tier mediates client requests across multiple independent database servers.

D - Incorrect: One centralized database server does not describe mediation across independent distributed servers.

Primary note proof

Study-note section ADB34

> In client/server architecture, clients submit requests while servers manage shared data, execute operations and enforce supported integrity/transaction rules. In collaborating-server designs, servers cooperate on subqueries and transactions. Middleware can translate requests, locate data, coordinate operations and combine results across independent servers. These describe communication responsibilities, not fragmentation types.

Question sources

Advanced Database Systems review questions · Question 49

Question 233 | NS0155

Hybrid fragmentation combines which two techniques?

A. Full replication and no replication

B. Horizontal fragmentation and vertical fragmentation

C. Locking and timestamp-based concurrency control

D. Client-server and middleware architectures

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer B: Horizontal and vertical fragmentation.

ADB36 explicitly defines hybrid fragmentation as combining row and column splits in an appropriate order. B names those two operations. Replication changes copy placement, locking and timestamps control concurrent access, and client-server/middleware describe architecture responsibilities. They can coexist with hybrid fragmentation but are not its two component fragmentation techniques.

Why each choice is right or wrong

A - Incorrect: Full/no replication concerns copies rather than the row and column splits of hybrid fragmentation.

B - Correct: Hybrid fragmentation combines horizontal row subsets with vertical column subsets.

C - Incorrect: Locking and timestamps are concurrency-control mechanisms, not fragmentation types.

D - Incorrect: These are architecture forms rather than the two data-fragmentation operations.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB36

> Hybrid fragmentation combines row and column splits in either suitable order.

Study-note section ADB34

> These describe communication responsibilities, not fragmentation types.

Related reading (context, not primary note proof)

Study-note section ADB37

Study-note section ADB11

Question sources

Advanced Database Systems review questions · Question 50

Question 234 | NS0286

In client-server database architecture, what is the primary role of the server?

A. Provide the GUI for end users

B. Initiate all network connections and poll clients for updates

C. Store and manage shared data, enforce ACID properties, and process client requests

D. Cache all user sessions locally for offline access

Hint

Identify which tier submits, mediates or executes a request.

Answer C: Manage shared data, process requests and enforce supported transaction/integrity guarantees.

ADB34 assigns clients request submission and servers shared-data management, operation execution and supported integrity/transaction enforcement. C matches those responsibilities. User interfaces usually belong to clients, while initiating all connections, polling every client or caching all sessions for offline use are not the server role defined by this architecture. Guarantees still depend on supported mechanisms and correct application use.

Why each choice is right or wrong

A - Incorrect: The user-facing GUI is normally a client responsibility rather than the database server's primary role.

B - Incorrect: The server need not initiate every connection or poll clients for every update.

C - Correct: Shared-data management, query execution and supported integrity/transaction enforcement are server duties.

D - Incorrect: Caching every session for offline access is not the defined database-server responsibility.

Primary note proof

Study-note section ADB34

> In client/server architecture, clients submit requests while servers manage shared data, execute operations and enforce supported integrity/transaction rules. In collaborating-server designs, servers cooperate on subqueries and transactions. Middleware can translate requests, locate data, coordinate operations and combine results across independent servers. These describe communication responsibilities, not fragmentation types.

Study-note section ADB34

> Calling a system heterogeneous does not require all sites to surrender autonomy or to copy every row everywhere.

Study-note section ADB34

> A tightly integrated distributed system can present one global schema and shared control.

Related reading (context, not primary note proof)

Study-note section ADB06

Question sources

Computer Science Exit Exam · 2018 · Question 3

Question 235 | NS0288

In two-phase locking (2PL), during which phase may a transaction acquire locks but not release any?

A. Growing phase

B. Commit phase

C. Validation phase

D. Shrinking phase

Hint

Track the first lock release and the transaction-completion boundary.

Answer A: Growing phase.

ADB12 allows acquiring or upgrading locks during growing and releasing or downgrading them during shrinking. Once any lock is released, a transaction may not acquire another. A is therefore the phase described. Commit is a transaction event, validation belongs to optimistic control and shrinking has the opposite lock-release role.

Why each choice is right or wrong

A - Correct: Growing permits acquisitions and upgrades before any release starts.

B - Incorrect: Commit is not the general 2PL phase for new lock acquisition.

C - Incorrect: Validation is an optimistic-control stage rather than the 2PL growing phase.

D - Incorrect: Shrinking releases or downgrades locks and forbids acquiring new ones.

Primary note proof

Study-note section ADB12

> Basic 2PL has a growing phase that acquires/upgrades locks and a shrinking phase that releases/downgrades them. Once any lock is released, no new lock may be acquired. The lock point is the instant the final lock is obtained. 2PL guarantees conflict serializability for well-formed transactions, but can deadlock and basic 2PL alone need not prevent cascading aborts.

Study-note section ADB12

> Two-phase locking orders access to data; two-phase commit coordinates a distributed commit decision.

Study-note section ADB16

> Optimistic/validation control reads and computes privately, validates its read/write sets against overlapping transactions, then publishes if validation succeeds; otherwise it retries.

Question sources

Computer Science Exit Exam · 2018 · Question 5

Question 236 | NS0306

Assume that we are able to specify that Mr. Getahun can view the database record, but cannot change the value of the record. Which security management technique was used in this scenario?

A. Access Control

B. Availability

C. Non-repudiation

D. Integrity

Hint

Identify whether the scenario concerns identity, permission, disclosure or valid values.

Answer A: Access control.

ADB24 explicitly describes a user allowed to view but not change a record as having restricted access privileges. That is authorization/access control. Integrity concerns valid values, availability concerns readiness for use and non-repudiation concerns attributable actions rather than whether a particular read or write is permitted. A matches the actual permission distinction.

Why each choice is right or wrong

A - Correct: Access control grants reading while denying modification for the specified user.

B - Incorrect: Availability concerns service readiness rather than the read-versus-write privilege distinction.

C - Incorrect: Non-repudiation does not define whether this user is authorized to modify a record.

D - Incorrect: Integrity restricts valid data states rather than assigning user-specific read/update permissions.

Primary note proof

Study-note section ADB24

> Authentication establishes identity; authorization/access control determines permitted actions. Encryption protects stored or transmitted content; integrity constraints restrict valid data; auditing records actions. A user who can view but cannot change a record has restricted access privileges. A view can restrict visible rows/columns, but its privilege and execution rules must support that restriction.

Study-note section ADB-S03

> Neither property specifies which particular user has permission to change a database row; that is the separate authorization decision.

Study-note section ADB-S03

> Availability concerns keeping services and resources accessible to their authorized users.

Question sources

Computer Science Exit Exam · 2018 · Question 27

Question 237 | NS0307

Suppose, a developer proposes using deferred update with no checkpoints in a high-transaction OLTP system. What is the most serious drawback?

A. Log scans become prohibitively expensive after long uptime

B. Increased disk I/O due to frequent page copying

C. Shared locks cannot be used

D. View serializability cannot be ensured

Hint

Identify which history a recovery boundary can safely exclude and whether active work spans it.

Answer A: Long-running log scans can make recovery expensive.

ADB30 warns that without checkpoints or another valid truncation boundary, a long-running system may scan expensive amounts of log history. Deferred update removes uncommitted persistent effects but does not remove that search problem. Page copying belongs to shadow approaches, while shared locks and serializability are independent concurrency concerns rather than consequences of omitting checkpoints.

Why each choice is right or wrong

A - Correct: High transaction volume and long uptime can produce costly recovery log scans without a valid boundary.

B - Incorrect: Frequent copy-on-write pages characterize shadow updating rather than the absence of checkpoints in deferred update.

C - Incorrect: Omitting checkpoints does not inherently prohibit shared locks.

D - Incorrect: View serializability is a concurrency-order property, not inherently lost by this checkpoint omission.

Primary note proof

Study-note section ADB30

> A checkpoint records enough durable recovery state to avoid scanning all old history. In a simple quiescent/checkpoint exercise, specified flushing may make earlier committed work unnecessary to redo. Active transactions can span the checkpoint, so their relevant earlier updates can still need undo. Do not assume every transaction before a checkpoint needs no processing.

Study-note section ADB30

> Without checkpoints or another valid truncation boundary, long-running systems may need expensive log scans; deferred update does not avoid that problem.

Study-note section ADB29

> Shadow/copy-on-write update writes a new location while preserving the committed version.

Related reading (context, not primary note proof)

Study-note section ADB11

Study-note section ADB09

Question sources

Computer Science Exit Exam · 2018 · Question 28

Question 238 | NS0321

Which listed techniques use transaction logs for recovery under the course model? Interpret Deferred Database Modification as conventional logged deferred update with no-force. Select all that apply.

A. Recovery in Multi-database Systems

B. ARIES Recovery Algorithm

C. Shadow Paging Technique

D. Deferred Database Modification

Hint

Separate update timing from update location and follow the commit boundary.

Answer B, D: B. ARIES Recovery Algorithm; D. Deferred Database Modification

ARIES relies on WAL and log-driven analysis, redo and undo. Conventional logged deferred update also uses recorded updates for redo of committed work, with no undo of uncommitted persistent changes. Both therefore belong to the requested category. Shadow paging preserves old page mappings instead, while recovery in multidatabase systems names an environment rather than a particular log-based technique.

Why each choice is right or wrong

A - Incorrect: Multidatabase recovery describes an environment rather than one specific recovery technique.

B - Correct: ARIES explicitly uses WAL and log-driven analysis, redo and undo.

C - Incorrect: Shadow paging preserves committed page maps rather than being the conventional log-based technique.

D - Correct: Deferred modification can use logged changes for redo after commitment.

Primary note proof

Study-note section ADB29

Deferred update keeps uncommitted changes outside the persistent database; in the usual no-force setting it is NO-UNDO/REDO. Redo-only and NO-UNDO/REDO mean the same thing, so those cannot be separate unique answers. Immediate update allows persistent changes before commit and therefore may require undo; it need not flush every write_item immediately.

Study-note section ADB31

ARIES uses WAL, page log sequence numbers (pageLSN), a transaction table and a dirty-page table.

Study-note section ADB29

It is an alternative to conventional log-based recovery, with allocation, fragmentation and concurrency costs; it is not primarily a special action done only during recovery.

Question sources

Computer Science Exit Exam · 2018 · Question 46

Question 239 | NS0348

Which type of fragmentation divides a relation into subrelations containing subsets of its columns, repeating a key or row identifier so the original rows can be reconstructed?

A. Replication

B. Location

C. Horizontal

D. Vertical

Hint

Identify whether rows or columns are split and reverse the split to reconstruct the original relation.

Answer D: D. Vertical

Vertical fragmentation divides a relation by columns. Repeating a key or row identifier allows corresponding pieces to be joined back into the original rows without spurious matches. Horizontal fragmentation divides rows instead. Replication creates copies, while location transparency hides where data resides. Hiding a column split would be fragmentation transparency, but the question asks for the split operation itself.

Why each choice is right or wrong

A - Incorrect: Replication copies data rather than dividing its columns.

B - Incorrect: Location transparency hides sites rather than splitting a relation.

C - Incorrect: Horizontal fragmentation selects row subsets.

D - Correct: Vertical fragmentation projects column subsets with a reconstruction identifier.

Primary note proof

Study-note section ADB36

> Horizontal fragmentation selects row subsets by predicates; their union reconstructs the relation. Check completeness so every original row is represented and disjointness if no overlap is intended. Define an explicit fragment for NULL/other values when needed. Vertical fragmentation projects column subsets and repeats a key or row identifier so joining on it reconstructs the original rows without spurious matches.

Study-note section ADB35

> Hiding this column split from queries is fragmentation transparency; the split itself is a data-design operation.

Study-note section ADB35

> Replication transparency hides which copy is used and how copies are maintained.

Related reading (context, not primary note proof)

Study-note section ADB37

Question sources

Computer Science Exit Exam · 2018 · Question 82

Object Oriented Programming

Questions & Worked Answers

294 questions + 4 source variants

Question 1 | Q00213

When overloading a function, what must be true?

A. The names should be different with different number and/or types of parameters.

B. None of the choices are correct

C. The names should be different with the same number and/or types of parameters.

D. The names should be the same with different number and/or types of parameters.

E. The names should be the same with the same number and/or types of parameters.

Hint

Compare names and parameter signatures before considering return types or access modifiers.

Answer D: The names should be the same with different number and/or types of parameters.

Overloads retain the method name and change parameter count or parameter types; return type alone cannot distinguish them.

Why each choice is right or wrong

A - Incorrect: Different names do not overload one function

B - Incorrect: D satisfies the overloading rule, so the statement that none of the choices is correct is false.

C - Incorrect: Different names define separate methods

D - Correct: Same name plus different parameter list meets the definition

E - Incorrect: Same name and same parameter signature cannot distinguish overloads

Primary note proof

Study-note section OOP12

> Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++. Overloads can also arise through inherited methods; “same class only” is a teaching simplification. f(int), f(double) and f(int,int) are distinct, but changing a parameter name does not create an overload.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 7-18

Method Overloading
• Is a process of having the same name for different methods as long as their parameter list or data types in the parameter list is different?
• The return type alone is insufficient to distinguish two versions of a method.
• Parameters are key identifiers of OM.
• Is one of the ways that Java supports polymorphism

Question sources

Mock exam 4 · Question 15

Question 2 | Q00983

Which of the following statement about function overloading is TRUE?

A. Overload function have same name but different parameters/type

B. Overload function is not supported

C. .
Overload function must have different return type

D. Overload function must have the same number of parameters

Hint

Compare names and parameter signatures before considering return types or access modifiers.

Answer A: Overload function have same name but different parameters/type

The test examples vary both argument counts and types while retaining test as the name.

Why each choice is right or wrong

A - Correct: Matches the definition

B - Incorrect: Overloading is explicitly supported

C - Incorrect: Return type alone is insufficient

D - Incorrect: test() and test(int,int) show that equal counts are not required

Primary note proof

Study-note section OOP12

Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++. Overloads can also arise through inherited

methods; “same class only” is a teaching simplification. f(int), f(double) and f(int,int) are distinct, but changing a parameter name does not create an overload.

Study-note section OOP12

The compiler selects the applicable signature from the declared argument types and conversions. int arguments usually choose an exact int overload rather than a wider double overload. A return type is not part of an ordinary Java method signature.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 7-18

Method Overloading
• Is a process of having the same name for different methods as long as their parameter list or data types in the parameter list is different?
• The return type alone is insufficient to distinguish two versions of a method.
• Parameters are key identifiers of OM.
• Is one of the ways that Java supports polymorphism

Object Oriented Programming module | PDF page 11 | lines 24-41

```
class OverloadDemo {
void test() {
System.out.println("No
parameters"); }
void test(int a) { // Overload
test for one integer parameter.
System.out.println("a: " + a); }
void test(int a, int b) { //
Overload test for two integer
parameters.
System.out.println("a and b: " +
a + " " + b); }
double test(double a) { //
overload test for a double
parameter
System.out.println("double a: "
+ a);
return a*a; } }
```

Question sources

Computer Science Exit Exam · 2018 · Question 96

Question 3 | Q01000

Which generation of programming languages stated for high-level languages like C++, Java, Python?

A. 2nd generation

B. 3rd generation

C. 4th generation

D. None of the above

Hint

Separate language-generation classification, translation strategy and portability assumptions.

Answer B: Third generation.

OOP39 explicitly places general-purpose high-level languages such as C++, Java and Python in the third generation. This taxonomy concerns historical language classification, not whether a language is object oriented or interpreted. Thus the third-generation option applies to all the examples named in the question.

Why each choice is right or wrong

A - Incorrect: Second generation is not the category assigned to these general-purpose high-level examples in OOP39.

B - Correct: Third generation is the explicit conventional category assigned to C++, Java and Python.

C - Incorrect: Fourth generation does not match the explicit third-generation classification of the named examples.

D - Incorrect: The third-generation option is present, so none of the above cannot be correct.

Primary note proof

Study-note section OOP39

> Under the conventional introductory generation taxonomy, general-purpose high-level languages such as C++, Java, Python and Perl are third generation. The taxonomy is separate from object orientation and from compilation versus interpretation. Java source normally compiles to JVM bytecode, and a compatible JVM plus libraries supports portability across systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

> The Java compiler translates the program
> into an intermediate language called Java
> bytecodes.

Java bytecodes are platform-independent.
Guaranteed to be Write Once, Run
Anywhere.
There are different Java interpreters for
different platforms.
Java is: + Object Oriented + Platform
Independent
+ Simple + Secure
+ Portable +
Multithreaded + Robust

Question sources

National Exit Exam practice questions · Question 14

Computer Science Exit Exam practice collection · Question 84

Question 4 | Q01143

In which generation that classifies and categorizes high-level language like C++, JAVA, Python, Perl.

A. a | First Generation Language x

B. i? Second Generation Language x

C. |@| Third Generation Language oS

D. | Fourth Generation Language x

Hint

Separate language-generation classification, translation strategy and portability assumptions.

Answer C: Third Generation Language.

C++, Java, Python and Perl are third-generation languages in this historical classification. Compilation versus interpretation and object orientation do not alter that historical classification.

Why each choice is right or wrong

A - Incorrect: First generation does not match the third-generation classification of the named high-level languages.

B - Incorrect: Second generation does not match the conventional category OOP39 assigns to these high-level examples.

C - Correct: The named general-purpose high-level languages are explicitly classified as third generation in OOP39.

D - Incorrect: Fourth generation is not the classification given for these examples in the complete note.

Primary note proof

Study-note section OOP39

> Under the conventional introductory generation taxonomy, general-purpose high-level languages such as C++, Java, Python and Perl are third generation. The taxonomy is separate from object orientation and from compilation versus interpretation. Java source normally compiles to JVM bytecode, and a compatible JVM plus libraries supports portability across systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

> The Java compiler translates the program
> into an intermediate language called Java
> bytecodes.
> Java bytecodes are platform-independent.
> Guaranteed to be Write Once, Run
> Anywhere.
> There are different Java interpreters for
> different platforms.
> Java is: + Object Oriented + Platform
> Independent
> + Simple + Secure
> + Portable +
> Multithreaded + Robust

Question sources

Computer Science Exit Exam practice questions · Question 4

Question 5 | SM0073

What is the output of the following Java fragment code? int n = 6, m = 15, p = 3; n -= m; n *= p; System.out.print(n);

A. 27

B. 3

C. 24

D. -27

Hint

Respect operator grouping and operand types, then trace each operation in evaluation order.

Answer D: -27.

Start with n=6. The subtraction assignment n-=m computes 6-15, leaving n=-9. The multiplication assignment then uses that updated value: -9*3=-27. OOP23 includes this exact trace. The sign cannot become positive because only subtraction followed by multiplication by positive three occurs.

Why each choice is right or wrong

A - Incorrect: 27 discards the negative sign produced when 15 is subtracted from 6.

B - Incorrect: Three is the unchanged multiplier p, not the updated value of n that is printed.

C - Incorrect: 24 does not follow either the subtraction result -9 or its subsequent multiplication by 3.

D - Correct: The ordered updates yield n=-9 and then n=-27, which System.out.print outputs.

Primary note proof

Study-note section OOP23

> Java evaluates operands left to right while respecting precedence. With n=6,m=15,p=3, n-=m then n*=p leaves -27.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Question sources

MOE Computer Science Exit Exam · Question 21-1

National Computer Science Exit Exam · Question 77

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 76

Question 6 | SM0098

What is the output of the following Java fragment code? int[] list = new int[4]; int sum = 0; for(int i = 0; i < list.length; i++) { list[i] = i * 3; sum += list[i]; } System.out.print(sum);

A. 18

B. 6

C. 3

D. 36

Hint

Trace every element or API argument and distinguish copying values from aliasing a reference.

Answer A: 18.

The length-four array is visited at indices 0,1,2,3. Assigning i*3 stores 0,3,6,9. Starting sum at zero and adding each element gives 18, exactly the example in OOP27. The loop processes four elements, not merely the first one or an uninitialized default array.

Why each choice is right or wrong

A - Correct: All four assigned elements contribute: 0+3+6+9 equals the printed sum 18.

B - Incorrect: Six is one element's value, not the sum after all four iterations.

C - Incorrect: Three is the element assigned at index 1, rather than the full accumulator result.

D - Incorrect: 36 doubles the actual total; each of the four elements is added only once.

Primary note proof

Study-note section OOP27

> For int[] a=new int[4], setting a[i]=i*3 and accumulating yields 0+3+6+9=18.

Study-note section OOP26

> Valid indices are 0..a.length-1; an invalid access throws ArrayIndexOutOfBoundsException.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)

• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Object Oriented Programming module | PDF page 7 | lines 26-41

Iteration: The for(), while() and do...while() statements
4. Jumping Statements: to jump out of loops and to control other areas of program java uses break and continue statements.
The break statement: used to exit a loop
The Continue Statement: used to force program to go back to the top of a loop
Labels: Java does not include a goto statement.
Instead of goto, Java allows you to combine break and continue with a label.

Question sources

MOE Computer Science Exit Exam · Question 28-1

National Computer Science Exit Exam · Question 42

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 79

Question 7 | SS178

Consider the following program:

```
import java.util.Scanner;
public class Test {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("enter choice");
int choice = scanner.nextInt();
switch (choice) {
case 1: System.out.println("Hello");
case 2: System.out.println("you");
break;
default: System.out.println("welcome");
}
System.out.println("Goodbye");
}
}
```

After entering 1, what lines are printed following the initial “enter choice” prompt? Preserve letter case.

A. Hello
You welcome

B. Hello
You
Welcome
Goodbye

C. Hello
Goodbye

D. Hello
you
Goodbye

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer D: Hello you Goodbye

Input 1 selects case 1, which prints Hello but has no break. Execution falls through to case 2 and prints you. Its break exits the switch before default, so welcome does not print. The final println outside the switch prints Goodbye. The requested output excludes the initial input prompt and preserves exact letter case.

Why each choice is right or wrong

A - Incorrect: This incorrectly includes welcome and omits the final Goodbye.

B - Incorrect: Default is skipped after case 2 breaks, so this includes an extra line.

C - Incorrect: Fall-through executes case 2, so omitting you is wrong.

D - Correct: The exact post-prompt lines are Hello, you and Goodbye.

Primary note proof

Study-note section OOP25

> break exits the loop; continue skips its remaining body and then runs a for update. Traditional colon-style switch labels fall through without break. Choice 1 printing Hello then case 2 printing you and breaking is followed by Goodbye outside the switch; default is skipped.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 51-70

> 2.3. Source file declaration
> rules
> • These rules are essential when declaring
> classes, import statements and package
> statements in a source file.
> 1. There can be only one public
> class per source file.
> 2. A source file can have multiple
> nonpublic classes.
> 3. The public class name should be
> the name of the source file as
> well which should be appended
> by .java at the end.
> For example: the class name is public class
> Employee{} then the source file should be
> as Employee.java.
> • If the class is defined inside a
> package, then the package statement
> should be the first statement in the
> source file.

Object Oriented Programming module | PDF page 7 | lines 26-41

> Iteration: The for(), while() and
> do...while() statements
> 4. Jumping Statements: to jump out
> of loops and to control other areas of
> program java uses break and
> continue statements.
> The break statement: used to exit a
> loop
> The Continue Statement: used to
> force program to go back to the top

of a loop
Labels: Java does not include a goto statement.
Instead of goto, Java allows you to combine break and continue with a label.

Question sources

Computer Science model exam collection · Question 85

Computer Science model exam collection

Question 8 | Q00035

Which one of the following is false regarding Exceptions in Java?

A. We cannot create our own exception classes by extending Exception class

B. Exception class describes errors caused by your program and external circumstances

C. Runtime Exception class describes programming errors such as bad casting and numeric errors

D. All Java exception classes inherit directly or indirectly from Throwable class

Hint

Separate source errors from runtime exceptions and determine the hierarchy and checked obligation.

Answer A: We cannot create our own exception classes by extending Exception class

The prohibition on user-defined exceptions is false: an exception class can extend Throwable or a subclass. Its examples describe input, file and resource failures and runtime programming bugs.

Why each choice is right or wrong

A - Correct: Contradicts the explicit permission to define exception classes

B - Incorrect: Consistent with user, programmer and resource failures

C - Incorrect: Consistent with runtime programming bugs, though individual bad-cast examples are not enumerated

D - Incorrect: Matches the Throwable ancestry rule

Primary note proof

Study-note section OOP29

Throwable is the root of throwable values, with Exception and Error branches.
RuntimeException and Error subclasses are unchecked; other throwable classes are checked.
IOException, FileNotFoundException and SQLException are standard checked examples.
ArithmeticException, NullPointerException, ClassCastException and array bounds exceptions are

unchecked. A missing semicolon or undeclared variable is a compile-time source error, not a checked exception.

Study-note section OOP29

A custom exception can extend Exception or RuntimeException.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 27 | lines 20-41

• An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.
• An exception is an indication of a problem that occurs during a program‟s execution.
• An exception is a runtime error.
• A Java exception is an instance of a class derived from Throwable.
• The Throwable class is contained in the java.lang package, and subclasses of Throwable are contained in various packages.
• Errors related to GUI components are included in the java.awt package; numeric exceptions are included in the java.lang package because they are related to the java.lang.Number class.
• You can create your own exception classes by extending Throwable or a subclass of Throwable.

Object Oriented Programming module | PDF page 27 | lines 42-54

• Exceptions occur for various reasons.
E.g.
• A user has entered invalid data.
• A file that needs to be opened cannot be found.
• A network connection has been lost in the middle of communications or the JVM has run out of memory.
• Some of these exceptions are caused by:
• User error,
• Programmer error,
• Physical resources that have failed in some manner.

Object Oriented Programming module | PDF page 28 | lines 65-79

✓ Unchecked exceptions: Is an exception that occurs at the time of execution, these are also called as Runtime Exceptions.
✓ These include programming bugs, such as logic errors or improper use of an API.
✓ Runtime exceptions are ignored at the time of compilation.
✓ For example, if you have declared an array of size 5 in your program, and trying to call the 6th element of the array then an ArrayIndexOutOfBoundsExceptione xception occurs.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 35

Mock exam 4 · Question 30

Question 9 | Q00067

Assuming there is a rectangle object, what getArea, getPerimeter can represent?

A. Attribute

B. Property

C. Data field

D. Behavior

Hint

Distinguish an object’s stored state, operations, identity and class definition.

Answer D: Behavior

getArea and getPerimeter describe operations performed by a rectangle, so they are behavior rather than its stored state.

Why each choice is right or wrong

A - Incorrect: An attribute stores state rather than performing this operation

B - Incorrect: Property here would mean stored/descriptive state, not the two operations

C - Incorrect: A data field stores a value; these are methods

D - Correct: Methods perform operations and express characteristic behavior

Primary note proof

Study-note section OOP01

An object has identity, state and behavior. State is stored in fields; behavior is provided by methods. A Rectangle object may hold width and height and answer getArea() or getPerimeter(). Two objects can have equal fields and still be distinct objects. A class defines their common structure and behavior; an object is a particular instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 2-18

• Are the basic runtime entities in
an OO program.
• Are identifiable things or
individuals?
• Are of significance to the system
• Have states reflected in the
attributes (data) held about
the object
• Have characteristic behaviour
(i.e. functionality)
ह. Class:
• Is a blueprint for an object.
• is the definition of a set of objects
• which are all similar in terms of their
attributes
• A class consists of a class name, data
and methods.

Object Oriented Programming module | PDF page 10 | lines 62-72

Methods
• Are members of classes and do
operations.
• Allows different classes to
communicate each other.
• E.g. type name(parameter-
list)
{ // body of method
//....
return value; // if
type is not void

Question sources

MOE Computer Science Exit Exam · 2016 · Question 67

Question 10 | Q00068

Which one of the following is false about abstract classes and abstract methods?

A. We can create instances of abstract classes using the new operator

B. Implementation of abstract methods is provided by the subclasses

C. A class that contains abstract methods must be defined as abstract

D. Abstract methods are defined without implementation in the superclass

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer A: We can create instances of abstract classes using the new operator

The abstract-class example permits a concrete method but forbids new A(); subclasses supply the body for callme(). Thus instantiation is the false claim.

Why each choice is right or wrong

A - Correct: Direct new instantiation is forbidden

B - Incorrect: B supplies the implementation of callme in the example

C - Incorrect: An abstract method is an unimplemented requirement of an abstract class in this model

D - Incorrect: The example declares abstract void callme() without a body

Primary note proof

Study-note section OOP18

> An abstract class cannot be directly instantiated but can define fields, constructors, abstract methods and concrete methods. It need not contain an abstract method. An abstract method has no body; abstract and final conflict because an operation cannot require overriding while forbidding it.

Study-note section OOP18

> A concrete subclass must provide or inherit implementations of all required abstract methods; otherwise it must itself be abstract.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 23 | lines 57-81

> Abstract Classes
> • A classes declared with abstract
> keyword and can‟t be instantiated.
> • Used to hide meaningless methods:
> • There can be no objects of an abstract

class. .:.
• An abstract class cannot be directly instantiated with the new operator.
• You cannot declare abstract constructors, or abstract static methods
• Syntax: abstract class class_name
Example:
abstract class A {
abstract void callme();
// concrete methods are still allowed in abstract classes
void callmetoo() {
System.out.println("This is a concrete method.");
}}
class B extends A {
void callme() {
System.out.println("B's implementation of callme.");
}}

Object Oriented Programming module | PDF page 24 | lines 63-73

Summery:
Abstract:
• Classes: declared with abstract keyword and can‟t be instantiated.
• Methods: without body (no implementation)
Final:
• Class: declared as final prevents inheritance.
• Method: declared as final prevents overriding that method.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 68

Question 11 | Q00070

Which operator is used to create an array?

A. new

B. declare

C. static

D. construct

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer A: new.

OOP26 shows int[] a=new int[5] as valid array creation. Here new is the allocation operator that creates the array object. The note also allows brace initializers without spelling new in declarations, so the answer identifies the operator rather than asserting that every array initializer must explicitly contain it.

Why each choice is right or wrong

A - Correct: new is the operator used in the note's explicit array-allocation form new int[5].

B - Incorrect: declare is not the allocation operator in the Java array-creation syntax shown in OOP26.

C - Incorrect: static controls class membership; it does not create an array object by itself.

D - Incorrect: construct is not the Java array-allocation operator used in the provided creation syntax.

Primary note proof

Study-note section OOP26

> int[] a=new int[5]; and int a[]={1,2,3}; are valid. int[] a={}; creates a zero-length array; explicit new is unnecessary in a declaration initializer.

Study-note section OOP07

> static gives one field per loaded class, shared by its instances.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 8 | lines 43-55

> • a class declaration only creates a template; it does not create an actual object.
> • To actually create a Dog object, use the second statement.
> • Dog myDog; // Creates only the template
> • myDog = new Dog(); // create a Dog object called myDog
> • After this statement executes, myDog will be an instance of Dog.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 70

Computer Science Exit Exam practice questions · Question 37

Question 12 | Q00072

What does this valid Java loop print? Ignore only the trailing space.

```
for (int i = 2; i <= 11; i = i + 2) {
if (i % 4 == 0)
System.out.print(i * i + " ");
}
```

A. 16 64

B. 4163664100

C. 16644864

D. 4816644

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer A: 16 64

The loop visits 2, 4, 6, 8 and 10. Only 4 and 8 have remainder zero when divided by 4, so their squares are the only values printed: 16 and 64. The string space separates the values. With valid Java capitalization, operators and parentheses, and ignoring the final space, the loop prints 16 64.

Why each choice is right or wrong

A - Correct: Only the qualifying values 4 and 8 print their squares, giving 16 64.

B - Incorrect: This includes squares from iterations where i is not divisible by four.

C - Incorrect: This adds values not printed by the two qualifying iterations.

D - Incorrect: This does not match the squares of 4 and 8.

Primary note proof

Study-note section OOP25

> while and for pre-test and may execute zero times; do-while post-tests and executes once.

Study-note section OOP23

> Java evaluates operands left to right while respecting precedence.

Study-note section OOP24

> Once a String operand is encountered, + concatenates in left-to-right groups: 1+2+"x" gives "3x", while "x"+1+2 gives "x12".

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Object Oriented Programming module | PDF page 7 | lines 26-41

> Iteration: The for(), while() and do...while() statements
> 4. Jumping Statements: to jump out of loops and to control other areas of program java uses break and continue statements.
> The break statement: used to exit a loop
> The Continue Statement: used to force program to go back to the top of a loop
> Labels: Java does not include a goto statement.

Instead of goto, Java allows you to combine break and continue with a label.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 72

Question 13 | Q00127

Which valid Java declaration reproduces a ragged table with first row 12, -9, 8; second row 7, 14; and third row -32, -1, 0? The second row has two elements, not a third element containing zero.

A. double[][] table = {12, -9, 8, 7, 14, -32, -1, 0};

B. double[][] table = { {12, -9, 8}, {7,14,0}, {-32,-1,0}};

C. double[][] table = { { 12,-9,8} {7,14} {-32,-1,0}};

D. double[][] table = { {12, -9, 8},{7, 14},{-32, -1, 0} };

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer D: Use rows {12,-9,8}, {7,14}, {-32,-1,0}.

Java represents a two-dimensional array as an array of row arrays, so the rows may have different lengths. The source table has lengths 3, 2 and 3. D supplies exactly those rows with comma-separated nested initializers. B is syntactically valid but adds an unwanted middle-row zero. A supplies scalar values where row arrays are needed, and C omits commas between its row initializers.

Why each choice is right or wrong

A - Incorrect: The declared double[][] requires row-array elements; a flat list of numeric scalars does not supply them.

B - Incorrect: Its second row contains an extra zero and therefore does not match the two-element row.

C - Incorrect: Missing commas between the row initializers make this declaration syntactically incorrect.

D - Correct: Its nested comma-separated rows match the required values and lengths 3, 2 and 3.

Primary note proof

Study-note section OOP26

int[] a=new int[5]; and int a[]={1,2,3}; are valid. int[] a={}; creates a zero-length array; explicit new is unnecessary in a declaration initializer.

Study-note section OOP26

Valid indices are 0..a.length-1; an invalid access throws ArrayIndexOutOfBoundsException. Java two-dimensional arrays are arrays of arrays and may be ragged: {{12,-9,8},{7,14},{-32,-1,0}} has row lengths 3,2,3. Java does not expose a portable physical contiguity guarantee, so a language-level answer should not rely on raw addresses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java
(byte, short, int, long, float, double,
char & Boolean)
2. Reference types: Arrays, classes,
and interfaces, String)
• Note: Boolean type has a value true or
false (there is no conversion b/n Boolean
and other types).
• All non-primitive types are reference
types, so all class types are reference
types.
• All integers in Java are signed, Java
doesn‟t support unsigned integers.

Question sources

Addis Ababa University mock exam · Question 29

Question 14 | Q00128

Which members are good candidates for static? PI is a shared mathematical constant; the salary display reads the receiver’s salary; the hypotenuse calculation uses only side-length parameters; each Student has its own name. Select all that apply.

A. The PI field of a circle class

B. A method that displays the salary of an Employee class

C. A method that calculates the hypotenuse of a right-angle triangle class

D. A name field of a Student class

E. All except B

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer A, C: The PI field of a circle class; A method that calculates the hypotenuse of a right-angle triangle class

PI is one shared mathematical constant, so a static final field suits it. A hypotenuse utility that uses only parameters also needs no current object and can be static. Reading an employee receiver's salary requires that instance, and each student's name belongs to its own object. The aggregate option includes the instance name field, so it is not a correct complete selection.

Why each choice is right or wrong

A - Correct: PI is shared across circles and is a suitable static final constant.

B - Incorrect: The specified display reads receiver-specific salary state and should be an instance method.

C - Correct: The specified calculation uses only parameters and is suitable as a static utility.

D - Incorrect: Each student's separate name should be an instance field.

E - Incorrect: All except B still includes the per-student name field, so this set is wrong.

Primary note proof

Study-note section OOP07

> static gives one field per loaded class, shared by its instances. If count starts at zero and every constructor increments it, three creations leave count=3. Prefer ClassName.count and ClassName.utility(). A static method has no current instance, but can still use instance members through an explicit object reference.

Study-note section OOP07

> A shared mathematical PI is a good static final candidate; a student name normally belongs to an instance. A calculation using only parameters may be static.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 5 | lines 39-61

> C. Class/Static variables
> • Are declared with the static keyword in a class, but outside a method, constructor or a block.
> • There would only be one copy of each class variable per class, regardless of how many objects are created from it.
> • Are rarely used other than being declared as constants. (public static final)
> • Are stored in the static memory.
> • Static variables are created when the program starts and destroyed

when the program stops.
• Visibility is similar to instance variables. However, most static variables are declared public
• Default values are same as instance variables.
• Static variables can be accessed by calling with the class name ClassName.VariableName.

Object Oriented Programming module | PDF page 14 | lines 2-19

Static methods and Final Variables
• A static method can access only static data.
• A static method can call only other static data (Static methods & variables), but can not call a non-static data from it.
• Static method is belongs to the class and not to the object.
• A static method can be accessed directly by the class name and doesn‟t need any object.
• i.e ClassName.methodName(args)
• A static method cannot refer to "this" or "super" keywords in anyway
• But can be overloaded since they are resolved using static binding by compiler at compile time.

Question sources

Addis Ababa University mock exam · Question 30

Question 15 | Q00129

Here is the general syntax for method definition:
accessModifier returnType methodName(parameterList){
Java statements
return returnValue;
}
What is true for the accessModifier?

A. It must always be private or public.

B. It can be omitted, but if not omitted it must be private or public.

C. It can be omitted, but if not omitted there are several choices, including private and public

D. The access modifier must agree with the type of the return value.

E. All of the above

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer C: It can be omitted, but if not omitted there are several choices, including private and public

The access table includes default/no modifier, private, public and protected. Access controls visibility and has no dependency on return type.

Why each choice is right or wrong

A - Incorrect: Excludes protected and default

B - Incorrect: Allows omission but incorrectly excludes protected

C - Correct: Allows omission and the several listed access levels

D - Incorrect: Return type and visibility have different roles

E - Incorrect: A, B and D are false

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible. Top-level classes/interfaces may be public or package-private, not protected/private. Member types have additional visibility possibilities.

Study-note section OOP11

> A method has modifiers, return type, name, parameter list and body: public int getAge(){return age;}.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

Access Specifiers
Access level modifiers determine whether other classes can use a particular field or invoke a particular method.
Their are four types access levels:
1. No Modifier(Default): Visible to the package(source file), and it is called package-private
2. Private: Visible to the class only.
3. Public: Visible to the world.
4. Protected: Visible to the package and all subclasses.

Question sources

Addis Ababa University mock exam · Question 31

Question 16 | Q00130

What is the output of the following Java 17 code?

```
class FinalExam{
int num1;
public FinalExam() {
num1 = 4;
}
void display() {
System.out.println(num1) ;
}
}
class Final1 extends FinalExam{
int num=3;
public Final1() {
this(5);
num1 = num;
}
public Final1(int num) {
this.num = num * num;
}
}
public class Main{
public static void main(String[] args){
Final1 f = new Final1();
f.display();
}
}
```

A. 4

B. 25

C. 12

D. 20

Hint

Trace constructor delegation and initialization in the specified Java version.

Answer B: 25

Constructing Final1 first runs the parent constructor, setting num1 to 4. Subclass initialization sets num to 3. The delegated Final1(int) body receives 5 and sets num to 25. Control returns to the no-argument body, which assigns num1=num, replacing 4 with 25. display then prints inherited num1. Consistent subclass and constructor names make this an executable Java 17 trace.

Why each choice is right or wrong

A - Incorrect: 4 is the parent's initial num1 value, overwritten by the no-argument subclass body.

B - Correct: The delegated constructor sets num=25 and the caller copies it to num1.

C - Incorrect: 12 is not produced by squaring the delegated argument or by the assignments.

D - Incorrect: 20 is not produced by the constructor chain.

Primary note proof

Study-note section OOP10

Under Java 17 rules, an explicit this(args) or super(args) invocation must be the first constructor statement. this invokes another constructor in the same class; super invokes the immediate superclass constructor. A constructor cannot explicitly invoke both. If there is no explicit delegation, an implicit super() is inserted; a matching accessible no-argument parent constructor must exist.

Study-note section OOP10

Trace parent construction first, then subclass field initializers and instance blocks in order, then the relevant subclass constructor body. With consistent class and constructor names and this(5) chaining in the FinalExam example, the parent sets num1=4, the subclass initializes num=3, this(5) sets num=25, and the no-arg body sets num1=num, leaving 25.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Object Oriented Programming module | PDF page 19 | lines 43-54

When Constructors Are Called?
• In a class hierarchy, constructors are called in order of derivation, from superclass to subclass.
• Further, since super( ) must be the first statement executed in a subclass‟ constructor, this order is the same whether or not super( ) is used.
• If super( ) is not used, then the default or parameter less constructor of each superclass will be executed.

Question sources

Addis Ababa University mock exam · Question 32

Question 17 | Q00131

Which one of the following is important to implement dynamic Polymorphism?

A. Polymorphic variable

B. Method overriding

C. Inheritance

D. All of the above

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer D: All of the above

Dynamic dispatch uses an inheritance hierarchy, an overridden method and a superclass reference pointing at a subclass object. The question's polymorphic variable is that reference.

Why each choice is right or wrong

A - Incorrect: The superclass reference can hold a child object

B - Incorrect: Overriding supplies the subclass behavior

C - Incorrect: Inheritance supplies the superclass/subclass relation

D - Correct: All three participate in the cited dispatch model

Primary note proof

Study-note section OOP15

For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Study-note section OOP14

A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype.

Study-note section OOP13

class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 22 | lines 37-53

• Dynamic method dispatch is the mechanism by which a call to an overridden method is resolved at run time, rather than compile time.
• Dynamic method dispatch is important because this is how Java implements run-time polymorphism
• When an overridden method is called through a superclass reference, Java determines which version of that method to execute based upon the type of the object being referred to at the time the call occurs.
• Thus, this determination is made at run time.
• Method Overriding is an example of runtime polymorphism.

Object Oriented Programming module | PDF page 25 | lines 17-37

• Polymorphism is the ability of an object to take on many forms.
• The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.
• Any Java object that can pass more than one IS-A test is considered to be polymorphic.
• In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class Object.

• Polymorphism is derived from 2 Greek words: poly and morphs.
• The word "poly" means many and "morphs" means forms. So polymorphism means many forms.
• polymorphism is the capability of an action or method to do different things based on the object that it is acting upon.

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Question sources

Addis Ababa University mock exam · Question 33

Question 18 | Q00132

An accessible parent declares public abstract int value(); and its direct child is non-abstract with no inherited implementation of that method. What must the child do?

A. A child must override an abstract method inherited from its parent by defining a method with the same signature and same return type

B. A child must define an additional method similar to the one inherited from its parent by defining a method with the same signature and different return type.

C. Achild must not define any method with the same signature as the parent's abstract method.

D. A non-abstract child must define an abstract method with the same signature and same return type as the parent's abstract method.

E. All of the above

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer A: A child must override an abstract method inherited from its parent by defining a method with the same signature and same return type

A concrete child must implement every outstanding abstract method. Here the inherited contract has no parameters and a primitive int return, so the child must supply a public concrete implementation of that same signature and return type. The covariant reference-return exception does not apply to primitive int. A different return type, refusing to implement the method, or declaring it abstract leaves the concrete-class requirement unsatisfied.

Why each choice is right or wrong

A - Correct: A concrete matching no-argument int implementation, with the required public access, satisfies the inherited contract.

B - Incorrect: A different return type cannot implement this primitive int-returning contract.

C - Incorrect: Omitting a matching implementation leaves an abstract obligation in the concrete child.

D - Incorrect: An abstract declaration supplies no implementation and cannot satisfy the concrete child’s obligation.

E - Incorrect: B, C and D fail the stated implementation requirement, so the aggregate cannot be correct.

Primary note proof

Study-note section OOP18

> A concrete subclass must provide or inherit implementations of all required abstract methods; otherwise it must itself be abstract. If parent declares abstract String welcome(), a String welcome() implementation is required; int welcome() is incompatible, and welcome(int) merely

adds an overload. A valid class can contain both the required implementation and an unrelated overload.

Study-note section OOP14

A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype. Visibility cannot become more restrictive, and checked exceptions cannot be broadened incompatibly.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 23 | lines 57-81

Abstract Classes
• A classes declared with abstract
keyword and can‟t be instantiated.
• Used to hide meaningless methods:
• There can be no objects of an abstract
class. .:.
• An abstract class cannot be directly
instantiated with the new operator.
• You cannot declare abstract
constructors, or abstract static methods
• Syntax: abstract class class_name
Example:
abstract class A {
abstract void callme();
// concrete methods are still
allowed in abstract classes
void callmetoo() {
System.out.println("This is a
concrete method.");
}}
class B extends A {
void callme() {
System.out.println("B's
implementation of callme.");
}}

Object Oriented Programming module | PDF page 11 | lines 55-67

Method Overriding
In a class hierarchy, when a method in a
subclass has the same name and type
signature as a method in its superclass, this
is called method Overriding.
• the method in the subclass is said to
override the method in the
superclass.

• The Method in the super class is known as an overridden method which is called within a subclass.
• The version of the method defined by the superclass will be hidden.

Object Oriented Programming module | PDF page 31 | lines 9-30

• It provides total abstraction;
• Means all the methods in interface are declared with empty body and are public and all fields are public, static and final by default.
• A class that implement interface must implement all the methods declared in the interface.
• If a class implements an interface and does not provide method bodies for all functions specified in the interface, then class must be declared abstract.
Rules of Interface
• Variables declared in an interface are public, static & final by default.
• Java does not allow you to extend more than one class, However you can implement more than one interfaces in your class.
• class implements interface but an interface extends another interfaces.

Question sources

Addis Ababa University mock exam · Question 34

Mock exam 3 · Question 18

Question 19 | Q00133

Which of the following is true?

A. A child class can extend a parent or implement an interface, but not do both.

B. A child class can extend just one parent and can implement just one interface.

C. A child class can extend just one parent and can implement zero or more interfaces.

D. A child class can extend zero or more parents, and can implement zero or more interfaces.

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer C: A child class can extend just one parent and can implement zero or more interfaces.

A Java class can extend one class and implement multiple interfaces.

Why each choice is right or wrong

A - Incorrect: No exclusive choice between extending and implementing is stated

B - Incorrect: Artificially limits interfaces to one

C - Correct: One direct superclass and zero or more interfaces matches the rule

D - Incorrect: Multiple direct parent classes are forbidden

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent. A class extends one direct class and can implement multiple interfaces. Single, multilevel and hierarchical inheritance mean one parent, a chain, and several children of one parent. Java does not permit class Dog extends Animal,Mammal.

Study-note section OOP19

> A class can extend a superclass and implement zero or more interfaces.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 31 | lines 9-30

> • It provides total abstraction;
> • Means all the methods in interface
> are declared with empty body and
> are public and all fields are public,
> static and final by default.
> • A class that implement interface
> must implement all the methods
> declared in the interface.
> • If a class implements an interface
> and does not provide method bodies
> for all functions specified in the
> interface, then class must be declared
> abstract.
> Rules of Interface
> • Variables declared in an interface are
> public, static & final by default.

• Java does not allow you to extend more than one class, However you can implement more than one interfaces in your class.
• class implements interface but an interface extends another interfaces.

Question sources

Addis Ababa University mock exam · Question 35

Mock exam 4 · Question 27

Question 20 | Q00224

What will be the output of the program?

```
try {
int x = 0;
int y = 5 / x;
} catch (Exception e) {
System.out.println("Exception");
} catch (ArithmeticException ae) {
System.out.println("Arithmetic Exception");
}
System.out.println("finished");
```

A. Arithmetic Exception

B. finished

C. Exception

D. Compilation fails.

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer D: Compilation fails because the subtype catch is unreachable.

Exception is caught before its subclass ArithmeticException. OOP30 says that a general catch before a subtype catch makes the latter unreachable and prevents compilation. The program therefore never executes the division or either print statement. Reasoning about which handler would run cannot bypass this compile-time restriction.

Why each choice is right or wrong

A - Incorrect: The ArithmeticException handler is unreachable at compile time, so it cannot print its message.

B - Incorrect: Compilation fails before execution; the statement following the catches never prints finished.

C - Incorrect: Although Exception could catch this runtime type, the illegal later catch prevents any execution.

D - Correct: The earlier Exception catch subsumes ArithmeticException, making the later handler unreachable and the program uncompilable.

Primary note proof

Study-note section OOP30

> Catch blocks are considered in source order. Place FileNotFoundException before IOException and IOException before Exception; a general catch before a subtype catch makes that later handler unreachable and prevents compilation.

Study-note section OOP29

> Throwable is the root of throwable values, with Exception and Error branches. RuntimeException and Error subclasses are unchecked; other throwable classes are checked. IOException, FileNotFoundException and SQLException are standard checked examples. ArithmeticException, NullPointerException, ClassCastException and array bounds exceptions are unchecked.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 29 | lines 11-22

> Using try and catch...
> General form of try/catch:
> Try{
> //block of code to monitor for
> errors
> }
> Catch(ExceptionType1 exOb) {
> //handler for ExceptionType1
> }
> Catch(ExceptionType2 exOb) {
> //handler for ExceptionType2
> }

Object Oriented Programming module | PDF page 27 | lines 20-41

> • An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.
> • An exception is an indication of a problem that occurs during a program‟s execution.
> • An exception is a runtime error.
> • A Java exception is an instance of a class derived from Throwable.

• The Throwable class is contained in the java.lang package, and subclasses of Throwable are contained in various packages.
• Errors related to GUI components are included in the java.awt package; numeric exceptions are included in the java.lang package because they are related to the java.lang.Number class.
• You can create your own exception classes by extending Throwable or a subclass of Throwable.

Question sources

Mock exam 4 · Question 26

Question 21 | Q00226

Which one of the following constant declaration statements is not possible in the Interface body?

A. static double PI = 3.14;

B. static final double PI = 3.14;

C. double PI = 3.14;

D. final double PI = 3.14;

E. None of the choices is correct

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer E: None of the choices is correct

All interface fields are implicitly public, static and final; writing either, both or neither of static and final does not remove the implicit modifiers. Each PI declaration also has an initializer.

Why each choice is right or wrong

A - Incorrect: static is explicit and final implicit

B - Incorrect: Both static and final are explicit

C - Incorrect: Both modifiers are implicit

D - Incorrect: final is explicit and static implicit

E - Correct: All four declarations satisfy the rule, so no listed declaration is impossible

Primary note proof

Study-note section OOP19

No interface instance fields or constructors exist; fields are implicitly public static final.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 31 | lines 9-30

• It provides total abstraction;
• Means all the methods in interface are declared with empty body and are public and all fields are public, static and final by default.
• A class that implement interface must implement all the methods declared in the interface.
• If a class implements an interface and does not provide method bodies for all functions specified in the interface, then class must be declared abstract.
Rules of Interface
• Variables declared in an interface are public, static & final by default.
• Java does not allow you to extend more than one class, However you can implement more than one interfaces in your class.
• class implements interface but an interface extends another interfaces.

Question sources

Mock exam 4 · Question 28

Question 22 | Q00227

Which one of the following is wrong about Method Overriding?

A. Method overriding is one way of implementing polymorphism at run time.

B. Method overriding deals with methods with similar signatures in the base and derived classes.

C. Method overriding deals with implementing similar operations for different objects.

D. Method overriding is possible between any two classes.

Hint

Check inheritance, signature compatibility, visibility and eligibility for an override.

Answer D: Method overriding is possible between any two classes.

Overriding is a superclass/subclass relationship with matching method signatures and supports runtime dispatch, not a relationship between arbitrary unrelated classes.

Why each choice is right or wrong

A - Incorrect: Runtime polymorphism is an explicit purpose

B - Incorrect: Matches the hierarchy and signature rule

C - Incorrect: Different subclass implementations can specialize the inherited operation

D - Correct: Unrelated arbitrary classes do not establish overriding

Primary note proof

Study-note section OOP14

> A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype.

Study-note section OOP14

> Merely reusing a name in two unrelated classes does not establish overriding.

Study-note section OOP15

> For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 55-67

> Method Overriding
> In a class hierarchy, when a method in a subclass has the same name and type signature as a method in its superclass, this is called method Overriding.
> • the method in the subclass is said to override the method in the superclass.
> • The Method in the super class is known as an overridden method which is called within a subclass.
> • The version of the method defined by the superclass will be hidden.

Object Oriented Programming module | PDF page 22 | lines 37-53

> • Dynamic method dispatch is the mechanism by which a call to an overridden method is resolved at run time, rather than compile time.

• Dynamic method dispatch is important because this is how Java implements run-time polymorphism
• When an overridden method is called through a superclass reference, Java determines which version of that method to execute based upon the type of the object being referred to at the time the call occurs.
• Thus, this determination is made at run time.
• Method Overriding is an example of runtime polymorphism.

Question sources

Mock exam 4 · Question 29

Question 23 | Q00277

Public classes A, B and C are in three different packages, with C extending B and B extending A. From code declared inside C, accessing fields through this, which is the complete access set? Assume ordinary instance fields with distinct names, accessible public declaring types, and no hiding.

A. All fields declared in C, and only accessible inherited protected and public fields from A and B

B. Protected, default and public data only in C and B.

C. Any data in C and only public data in A and B.

D. Protected, default and public data only in C.

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer A: All fields declared in C, and only accessible inherited protected and public fields from A and B

Code inside C can access C’s own fields, including private ones. Across package boundaries it can also access inherited public and protected fields through its own inherited context, here represented by this. It cannot access package-private or private ancestor fields directly. Consequently A gives the complete set; the public-only choice is incomplete because qualifying protected inherited access is also permitted.

Why each choice is right or wrong

A - Correct: This includes C’s own fields and the inherited public/protected access permitted through this.

B - Incorrect: Package-private ancestor fields in different packages are not accessible, and A's inherited access is omitted.

C - Incorrect: This omits accessible inherited protected fields and therefore is not the complete set.

D - Incorrect: This omits C's own private access and permitted inherited ancestor access.

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible. Top-level classes/interfaces may be public or package-private, not protected/private. Member types have additional visibility possibilities.

Study-note section OOP08

> Across packages, a subclass accessing a protected instance member must do so through its own inherited context or an appropriate subclass-typed receiver, not any arbitrary superclass object.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether
> other classes can use a particular field or
> invoke a particular method.
> Their are four types access levels:
> 1. No Modifier(Default): Visible to
> the package(source file), and it is
> called package-private
> 2. Private: Visible to the class only.
> 3. Public: Visible to the world.
> 4. Protected: Visible to the package
> and all subclasses.

Object Oriented Programming module | PDF page 17 | lines 4-9

> Member Access and Inheritance
> • Although a subclass includes all of
> the members of its superclass, it
> cannot access those members of the
> superclass that have been declared as
> private.

Question sources

Computer Science model exam · Question 8

University of Gondar model exam · Question 8

Question 24 | Q00278

Inside an instance method, which shown forms invoke an accessible no-argument non-static method? Methodname stands for a valid method name, Objectname for a compatible object reference, and Classname for a type name. Select all that apply.

A. Methodname()

B. Classname.methodname()

C. Objectname.methodname()

D. Variablename.methodname

Hint

Separate parameters from arguments and operations that read state from operations that change it.

Answer A, C: Methodname(); Objectname.methodname()

An unqualified call inside a suitable instance context uses implicit this, so Methodname() is valid. An object-qualified call explicitly supplies the receiver and is also valid. A type name alone cannot invoke a non-static method without an object. The final form lacks invocation parentheses and therefore does not invoke a method. These are deliberate syntax alternatives under the stated placeholder meanings.

Why each choice is right or wrong

A - Correct: The unqualified call uses implicit this in the stated instance context.

B - Incorrect: A class name alone does not provide the required instance receiver.

C - Correct: The object-qualified call supplies the instance receiver.

D - Incorrect: Without parentheses this form does not perform a method invocation.

Primary note proof

Study-note section OOP11

> A method has modifiers, return type, name, parameter list and body: public int getAge(){return age;}. A void method returns no value. Parameters name the inputs in the declaration; arguments supply values at a call. An instance method can be invoked as object.method(), or with an implicit this inside suitable instance context.

Study-note section OOP16

> Java this and super are unavailable as current-instance references in static context.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 14 | lines 2-19

Static methods and Final Variables
• A static method can access only static data.
• A static method can call only other static data (Static methods & variables), but can not call a non-static data from it.
• Static method is belongs to the class and not to the object.
• A static method can be accessed directly by the class name and doesn‟t need any object.
• i.e ClassName.methodName(args)
• A static method cannot refer to "this" or "super" keywords in anyway
• But can be overloaded since they are resolved using static binding by compiler at compile time.

Object Oriented Programming module | PDF page 11 | lines 24-41

```
class OverloadDemo {
void test() {
System.out.println("No
parameters"); }
void test(int a) { // Overload
test for one integer parameter.
System.out.println("a: " + a); }
void test(int a, int b) { //
Overload test for two integer
parameters.
System.out.println("a and b: " +
a + " " + b); }
double test(double a) { //
overload test for a double
parameter
System.out.println("double a: "
+ a);
return a*a; } }
```

Question sources

Computer Science model exam · Question 9

University of Gondar model exam · Question 9

Question 25 | Q00279

In Java, Base declares an eligible instance method m(), Sub overrides it, and an object's exact runtime class is Sub. Inside an instance method declared in Sub, what happens on an unqualified call m()? Assume no further subclass or explicit super call.

A. The Compiler will choose randomly the method to be executed

B. It will always refer to the version of that method defined by the subclass

C. Compile time error will occur

D. It will always refer to the version of that method defined by the superclass

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer B: It will always refer to the version of that method defined by the subclass

The unqualified invocation uses the current Sub object as receiver. Since m is an eligible overridden instance method and the exact runtime class is Sub, dynamic dispatch selects Sub's implementation. The compiler does not choose randomly, and the valid declaration produces no compilation error. An explicit super.m() would request the parent version, but that is excluded by the specified call expression.

Why each choice is right or wrong

A - Incorrect: Dispatch follows the language's receiver and overriding rules, not random choice.

B - Correct: The exact Sub receiver dispatches the ordinary call to Sub.m().

C - Incorrect: The stated accessible, eligible declaration and call compile.

D - Incorrect: The superclass version would require a different dispatch situation such as explicit super.m().

Primary note proof

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Study-note section OOP16

> An unqualified instance call may dispatch dynamically; an explicit super call is different.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 55-67

> Method Overriding
> In a class hierarchy, when a method in a
> subclass has the same name and type

signature as a method in its superclass, this is called method Overriding.
• the method in the subclass is said to override the method in the superclass.
• The Method in the super class is known as an overridden method which is called within a subclass.
• The version of the method defined by the superclass will be hidden.

Object Oriented Programming module | PDF page 17 | lines 51-74

The super keyword
• Since encapsulation is a primary attribute of OOP, Whenever a subclass needs to refer to its immediate superclass, it can do so by use of the keyword super.
• Super has two general forms.
1. Used to access parent class instance members.
2. Used to access parent class constructor.
1. Used to access parent class instance members.
• super acts somewhat like this, except that it always refers to the superclass of the subclass in which it is used.
• Super has the form: super.member Here, member can be either a method or an instance variable.
• Super is most applicable to resolve member hiding.
• super( ) must always be the first statement executed inside a subclass‟ constructor.

Question sources

Computer Science model exam · Question 10

University of Gondar model exam · Question 10

Question 26 | Q00280

Which of the following statements not true?

A. A private member of a class cannot be accessed by the methods of the same class

B. A public member of a class can be accessed in all the packages

C. A private member of a class cannot be accessed from its derived class

D. A protected member of a class can be accessed from its derived class.

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer A: A private member of a class cannot be accessed by the methods of the same class

Private visibility is class-only, so methods of the declaring class can access that member. Public is world-visible; protected includes subclasses.

Why each choice is right or wrong

A - Correct: False: same-class methods have access

B - Incorrect: Public visibility includes other packages

C - Incorrect: A child cannot directly access a parent's private member

D - Incorrect: Protected includes subclass access

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible. Top-level classes/interfaces may be public or package-private, not protected/private. Member types have additional visibility possibilities.

Study-note section OOP08

> Across packages, a subclass accessing a protected instance member must do so through its own inherited context or an appropriate subclass-typed receiver, not any arbitrary superclass object. protected therefore does not mean “subclasses only”: unrelated classes in the same package also have access. A class’s own main method can access private fields of an instance of that class.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether
> other classes can use a particular field or
> invoke a particular method.
> Their are four types access levels:

1. No Modifier(Default): Visible to the package(source file), and it is called package-private
2. Private: Visible to the class only.
3. Public: Visible to the world.
4. Protected: Visible to the package and all subclasses.

Object Oriented Programming module | PDF page 17 | lines 4-9

Member Access and Inheritance
• Although a subclass includes all of the members of its superclass, it cannot access those members of the superclass that have been declared as private.

Question sources

Computer Science model exam · Question 11

University of Gondar model exam · Question 11

Question 27 | Q00281

_____is one of the basic principles of object oriented programming that use one object in multiple forms.

A. Inheritance

B. Interface

C. Encapsulation

D. None of the above

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer D: None of the above

Taking many forms is polymorphism. It is not explicitly listed, so the none option is the correct selection.

Why each choice is right or wrong

A - Incorrect: Inheritance acquires parent properties

B - Incorrect: Interface defines a contract rather than naming the many-forms principle

C - Incorrect: Encapsulation groups and hides data

D - Correct: The principle is polymorphism

Primary note proof

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Study-note section OOP02

> Encapsulation groups related state and operations and controls access.

Study-note section OOP19

> interface Payable{double getPaymentAmount();} declares an abstract public operation. class Invoice implements Payable must provide public double getPaymentAmount(), perhaps returning getQuantity()*getPricePerItem().

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 58-64

> 2. Polymorphism:
> • The word polymorphism means the ability to take more than one form.
> • In terms of the OOP, this means that a particular operation may behave differently for different sub-classes of the same class.

Question sources

Computer Science model exam · Question 12

University of Gondar model exam · Question 12

Question 28 | Q00282

Consider the program:

public class Student {

int id;

int section;

void display() {

System.out.println("ID : " + id);
System.out.println("Section : " + section); }

public static void main(String[] args) {

/*Create an object*/ }}

Which of the following is a correct way to create an object in the main method of the above program?

A. Student stud = Student (3983,20);

B. Student stud;

C. Student stud=new Student ();

D. Student stud =new Student (3983,2);

Hint

Separate a reference declaration, object allocation, reference copying and changes to the shared object.

Answer C: Student stud=new Student ();

No constructor is declared in Student, so the default no-argument construction pattern applies. A reference declaration alone does not create an object.

Why each choice is right or wrong

A - Incorrect: Missing new and invokes an undeclared parameterized constructor

B - Incorrect: Declares only a reference

C - Correct: Uses new and the implicit no-argument constructor

D - Incorrect: No matching two-argument constructor is declared

Primary note proof

Study-note section OOP03

> Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Study-note section OOP09

If a class declares no constructor, Java supplies a default constructor. If Circle(double r) is declared, new Circle() fails unless a no-argument constructor is also declared.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 8 | lines 43-55

• a class declaration only creates a template; it does not create an actual object.
• To actually create a Dog object, use the second statement.
• Dog myDog; // Creates only the template
• myDog = new Dog(); // create a Dog object called myDog
• After this statement executes, myDog will be an instance of Dog.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Question sources

Computer Science model exam · Question 13

University of Gondar model exam · Question 13

Question 29 | Q00431

Default, public , protected, private are?

A. Access Modifier

B. Non-access Modifiers

C. Both A and B

D. It's variable

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer A: Access Modifier

The access table names no modifier/default, public, protected and private as visibility levels.

Why each choice is right or wrong

A - Correct: They control accessibility

B - Incorrect: They are visibility modifiers rather than non-access properties such as static

C - Incorrect: No basis for classifying the same list as both

D - Incorrect: They are not a variable

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible. Top-level classes/interfaces may be public or package-private, not protected/private. Member types have additional visibility possibilities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether
> other classes can use a particular field or
> invoke a particular method.
> Their are four types access levels:
> 1. No Modifier(Default): Visible to
> the package(source file), and it is

called package-private
2. Private: Visible to the class only.
3. Public: Visible to the world.
4. Protected: Visible to the package
and all subclasses.

Question sources

Object-Oriented Programming practice questions · Question 1

Question 30 | Q00432

Java Source Code is compiled into _____.

A. .Obj

B. Source Code

C. Bytecode

D. .Exe

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer C: Bytecode

The compiler produces intermediate Java bytecodes, explicitly described as platform-independent.

Why each choice is right or wrong

A - Incorrect: Not the compiler’s intermediate-language output

B - Incorrect: Input source is translated rather than simply reproduced

C - Correct: Matches the stated intermediate language

D - Incorrect: Not the compiler’s intermediate-language output

Primary note proof

Study-note section OOP28

Ordinary compilation uses javac Test.java, producing bytecode, then java Test launches the class.

Study-note section OOP39

Java source normally compiles to JVM bytecode, and a compatible JVM plus libraries supports portability across systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

The Java compiler translates the program into an intermediate language called Java bytecodes.
Java bytecodes are platform-independent.
Guaranteed to be Write Once, Run Anywhere.
There are different Java interpreters for different platforms.
Java is: + Object Oriented + Platform Independent
+ Simple + Secure
+ Portable +
Multithreaded + Robust

Question sources

Object-Oriented Programming practice questions · Question 2

Question 31 | Q00433

Single line comment starts with _____ in Java.

A. /**

B. //

C. /*

D. None of these

Hint

Check the lexical category and whether the notation names a value, a declaration or non-executed source text.

Answer B: //

// begins a single-line comment; /* and /** begin block comments.

Why each choice is right or wrong

A - Incorrect: Starts documentation-style block comments

B - Correct: The single-line delimiter

C - Incorrect: Starts a block comment

D - Incorrect: The required delimiter is listed as B

Primary note proof

Study-note section OOP-S01

> A single-line Java comment begins with // and continues to the end of that line. A block comment begins with /* and ends with */. A documentation comment begins with /** and ends with */ and can be processed by Javadoc. Comment text does not execute as program statements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 2-4

> single line comment //, 2 multiline
> comments /* .... */ and /** ....*/.
> The las one is used by javaDoc

Question sources

Object-Oriented Programming practice questions · Question 3

Question 32 | Q00434

How to compile java code in command prompt?

A. javac filename.java

B. java filename.java

C. javac filename

D. java filename

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer A: javac filename.java

The FileNotFound example shows C:\>javac FilenotFound_Demo.java, establishing the compiler command with a .java source filename.

Why each choice is right or wrong

A - Correct: Matches the javac compilation command

B - Incorrect: Uses java rather than the demonstrated compiler

C - Incorrect: Omits the .java source suffix

D - Incorrect: Uses the runtime command name rather than javac

Primary note proof

Study-note section OOP28

> Ordinary compilation uses javac Test.java, producing bytecode, then java Test launches the class. Modern Java can also launch a source file using java Test.java; distinguish source-file launch from running a compiled class.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 28 | lines 54-64

> • If you try to compile the
> above program you will get
> exceptions like this.
> C:\>javac FilenotFound_Demo.java
> FilenotFound_Demo.java:8: error:
> unreported exception
> FileNotFoundException; must be
> caught or declared to be thrown
> FileReader fr = new
> FileReader(file);
> 1 error

Question sources

Object-Oriented Programming practice questions · Question 4

Question 33 | Q00435

An ordinary class named filename has already been compiled to filename.class, is in the current directory with no package, and has a valid main method. Which command launches that compiled class? Use class launch, not source-file mode.

A. javac filename.java

B. java filename.java

C. javac filename

D. java filename

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer D: java filename

The Java launcher takes the compiled class's name for ordinary class launch, so java filename runs the available main method. javac filename.java compiles a source file and is a separate step. Modern Java also supports source-file launch with a .java argument, but the revised question explicitly asks for the already compiled class. The compiler invocation without the source extension is not the requested launch either.

Why each choice is right or wrong

A - Incorrect: This compiles source rather than launching the existing compiled class.

B - Incorrect: This is source-file launch, excluded by the requested class-launch mode.

C - Incorrect: This is a compiler invocation rather than the requested class launch.

D - Correct: java filename launches the named compiled class.

Primary note proof

Study-note section OOP28

> Ordinary compilation uses javac Test.java, producing bytecode, then java Test launches the class. Modern Java can also launch a source file using java Test.java; distinguish source-file launch from running a compiled class.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

> The Java compiler translates the program
> into an intermediate language called Java
> bytecodes.

Java bytecodes are platform-independent.
Guaranteed to be Write Once, Run
Anywhere.
There are different Java interpreters for
different platforms.
Java is: + Object Oriented + Platform
Independent
+ Simple + Secure
+ Portable +
Multithreaded + Robust

Object Oriented Programming module | PDF page 28 | lines 54-64

• If you try to compile the
above program you will get
exceptions like this.
C:\>javac FilenotFound_Demo.java
FilenotFound_Demo.java:8: error:
unreported exception
FileNotFoundException; must be
caught or declared to be thrown
FileReader fr = new
FileReader(file);
1 error

Question sources

Object-Oriented Programming practice questions · Question 5

Question 34 | Q00436

Java is case sensitive?

A. True

B. False

C. Depends On Complier

D. May be true or false

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer A: True: Java is case sensitive.

OOP28 explicitly states that Java identifiers are case sensitive. Changing the capitalization of a name can therefore change what it denotes or make a reference unresolved. This is a language rule rather than a compiler preference, so the conditional or uncertain answers do not describe standard Java behavior.

Why each choice is right or wrong

A - Correct: True directly matches the note's explicit statement that Java identifiers are case sensitive.

B - Incorrect: False contradicts the explicit identifier case-sensitivity rule stated in OOP28.

C - Incorrect: Case sensitivity is a Java language rule, not an option depending on the compiler implementation.

D - Incorrect: The note states a definite language property rather than a true-or-false implementation-dependent possibility.

Primary note proof

Study-note section OOP28

> Java identifiers are case sensitive; $ is allowed but @ is not an identifier character.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 11-20

> ✓ Reserved keywords: class, extends, implements, for... ✓ Identifiers: programmers given name; grade, gRade... ✓ Literals (Constants values): 55, 60.78, "Hi", „M"... Java language specifies five major types of literals => integer, floating, character, string & Boolean literals

Question sources

Object-Oriented Programming practice questions · Question 6

Question 35 | Q00437

Which is the most complete description of the standard introductory naming conventions and ordinary public-top-level-class file rule in Java? Treat "should" in A and B as style advice, not a syntax requirement, and "class" in C as the public top-level class in a .java source file.

A. For all class names the first letter should be in Upper Case.

B. All method names should start with a Lower Case letter.

C. Name of the program file should exactly match the class name.

D. All of the above

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer D: All of the above

Capitalized class initials and lowercase method initials are conventional Java style, not universal syntax requirements. An ordinary public top-level class must match its source filename's base name. The scope makes A and B recommendations and C the specific file rule. Each is then true, but D is the most complete selection because it includes all three rather than one isolated part.

Why each choice is right or wrong

A - Incorrect: This is valid style advice but only one part of the complete set.

B - Incorrect: This is valid style advice but omits the other convention and file rule.

C - Incorrect: This is the file rule for the specified public top-level class but is incomplete alone.

D - Correct: All three are valid under the stated convention and file-rule scope.

Primary note proof

Study-note section OOP28

> For ordinary public top-level types, filename and type name match. Capitalized class names and lowercase method names are conventions, not universal syntax requirements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 51-70

> 2.3. Source file declaration
> rules
> • These rules are essential when declaring
> classes, import statements and package
> statements in a source file.
> 1. There can be only one public
> class per source file.
> 2. A source file can have multiple
> nonpublic classes.
> 3. The public class name should be
> the name of the source file as
> well which should be appended
> by .java at the end.
> For example: the class name is public class
> Employee{} then the source file should be
> as Employee.java.
> • If the class is defined inside a
> package, then the package statement
> should be the first statement in the
> source file.

Object Oriented Programming module | PDF page 4 | lines 11-20

✓ Reserved keywords: class, extends, implements, for...
✓ Identifiers: programmers given name; grade, gRade...
✓ Literals (Constants values): 55, 60.78, “Hi”, „M“... Java language specifies five major types of literals => integer, floating, character, string & Boolean literals

Question sources

Object-Oriented Programming practice questions · Question 7

Question 36 | Q00438

All Java components require names. Names used for classes, variables, and methods are called?

A. Variables

B. identifiers

C. Access Modifiers

D. Java Modifiers

Hint

Check the lexical category and whether the notation names a value, a declaration or non-executed source text.

Answer B: identifiers

Identifiers are programmer-given names. Names attached to classes, variables and methods serve that role.

Why each choice is right or wrong

A - Incorrect: A variable is one named entity, not the name category

B - Correct: Matches programmer-given names

C - Incorrect: Controls visibility rather than naming

D - Incorrect: Modifiers change declarations rather than supply their names

Primary note proof

Study-note section OOP-S01

An identifier is a programmer-chosen name for an entity such as a class, method or variable. A reserved keyword has a language-defined role and cannot be used as an ordinary identifier. A

literal writes a value directly, such as an integer, floating-point number, quoted character, quoted string or Boolean value.

Study-note section OOP08

For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 11-20

✓ Reserved keywords: class, extends, implements, for...
✓ Identifiers: programmers given name; grade, gRade...
✓ Literals (Constants values): 55, 60.78, “Hi”, „M“... Java language specifies five major types of literals => integer, floating, character, string & Boolean literals

Question sources

Object-Oriented Programming practice questions · Question 8

Question 37 | Q00439

Which of the following is smallest integer data type ?

A. int

B. byte

C. short

D. long

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer B: byte

The actual PDF page-six widening diagram orders byte -> short -> int -> long -> float (and onward to double); the text says widening requires the destination to be wider than the source. Among byte, short, int and long, byte is therefore the smallest in this module diagram. The diagram is a raster/embedded visual omitted from extracted text, so the exact textual citation supplies the associated width condition rather than fabricating a diagram quotation.

Why each choice is right or wrong

A - Incorrect: int appears after byte and short in the widening diagram

B - Correct: byte is the earliest/narrowest listed integer type in the diagram

C - Incorrect: short is wider than byte in the diagram

D - Incorrect: long appears after int in the diagram

Primary note proof

Study-note section OOP06

> Java has byte, short, int, long, float, double, char and boolean. Signed integer widths are 8,16,32,64 bits respectively. byte ranges from -128 to 127; short from -32768 to 32767.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean
> and other types).
> • All non-primitive types are reference
> types, so all class types are reference
> types.
> • All integers in Java are signed, Java
> doesn‟t support unsigned integers.

Object Oriented Programming module | PDF page 6 | lines 30-36

> Two conditions for automatic
> casting
> ✓ Types are compatible
> ✓ Destination is wider
> than the source
> Widening conversion usually
> occur automatically.

Question sources

Object-Oriented Programming practice questions · Question 9

Question 38 | Q00440

Which of the following is not a primitive data type ?

A. byte

B. enum

C. short

D. int

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer B: enum

The exhaustive eight-primitive list includes byte, short and int but not enum.

Why each choice is right or wrong

A - Incorrect: Explicit primitive

B - Correct: Not among the eight primitives

C - Incorrect: Explicit primitive

D - Incorrect: Explicit primitive

Primary note proof

Study-note section OOP06

> Java has byte, short, int, long, float, double, char and boolean.

Study-note section OOP06

> enum and String are reference types.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean
> and other types).
> • All non-primitive types are reference
> types, so all class types are reference

types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 10

Question 39 | Q00441

Integer Data type does not include following primitive data type _____.

A. long

B. byte

C. short

D. double

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer D: double is not an integer primitive type.

OOP06 lists byte, short, int and long as the signed integer types with widths 8,16,32,64 bits. Of the supplied options, long, byte and short belong to that group. double represents floating-point values rather than an integer primitive, so it is the exception requested by the question.

Why each choice is right or wrong

A - Incorrect: long is the 64-bit signed integer primitive in the note's integer-width sequence.

B - Incorrect: byte is the 8-bit signed integer primitive with the stated range -128 to 127.

C - Incorrect: short is the 16-bit signed integer primitive with the stated range -32768 to 32767.

D - Correct: double belongs to the floating numeric types rather than the signed integer primitive group.

Primary note proof

Study-note section OOP06

Java has byte, short, int, long, float, double, char and boolean. Signed integer widths are 8,16,32,64 bits respectively.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 11

Question 40 | Q00442

Character data type cannot store following value.

A. Digit

B. Letter

C. Special Character

D. String

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer D: String

Character literals such as 'M' represent individual characters, while strings such as "Hi" are sequences of characters. A whole String is not one char value.

Why each choice is right or wrong

A - Incorrect: A digit can be a single character, not a multi-character string

B - Incorrect: A letter is a single character, illustrated by M

C - Incorrect: A special symbol can be one character

D - Correct: A string is a sequence rather than one character

Primary note proof

Study-note section OOP06

char is an unsigned 16-bit UTF-16 code unit and can hold a digit, letter or symbol, not a whole String.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 11-20

✓ Reserved keywords: class, extends, implements, for...
✓ Identifiers: programmers given name; grade, gRade...
✓ Literals (Constants values): 55, 60.78, “Hi”, „M“... Java language specifies five major types of literals => integer, floating, character, string & Boolean literals

Object Oriented Programming module | PDF page 9 | lines 14-16

5. String Class:- Immutable Class created to manipulate strings(Sequence of chars).

Question sources

Object-Oriented Programming practice questions · Question 12

Question 41 | Q00443

Range of Byte Data Type is _____.

A. -128 to 128

B. -127 to 127

C. -127 to 128

D. -128 to 127

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer D: -128 to 127, inclusive.

OOP06 states that byte ranges from -128 to 127. These inclusive endpoints provide 256 values for an 8-bit signed type. Options that include positive 128 exceed the upper endpoint, and those starting at -127 omit the lowest permitted value. Only D preserves both limits correctly.

Why each choice is right or wrong

A - Incorrect: The upper endpoint 128 is one greater than the largest byte value 127.

B - Incorrect: The lower endpoint -127 omits the valid minimum byte value -128.

C - Incorrect: Both endpoints are wrong: -128 is omitted and the invalid positive value 128 is included.

D - Correct: The inclusive limits -128 and 127 exactly match the byte range stated in OOP06.

Primary note proof

Study-note section OOP06

> byte ranges from -128 to 127; short from -32768 to 32767.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean
> and other types).
> • All non-primitive types are reference
> types, so all class types are reference
> types.
> • All integers in Java are signed, Java
> doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 13

Question 42 | Q00444

What is size of integer in Java Programming.

A. 1 Bytes

B. 2 Bytes

C. 4 Bytes

D. 8 Bytes

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer C: 4 bytes (32 bits).

In OOP06's ordered list, int is the 32-bit signed integer type. Converting 32 bits at eight bits per byte gives four bytes. The question concerns Java's fixed int width rather than implementation-dependent C++ sizes. The one-, two- and eight-byte options correspond to different integer widths.

Why each choice is right or wrong

A - Incorrect: One byte corresponds to the 8-bit byte type, not the 32-bit Java int.

B - Incorrect: Two bytes correspond to the 16-bit short width, not the Java int width.

C - Correct: Java int has 32 bits, equivalent to four eight-bit bytes.

D - Incorrect: Eight bytes correspond to the 64-bit long width, rather than the Java int width.

Primary note proof

Study-note section OOP06

> Signed integer widths are 8,16,32,64 bits respectively.

Study-note section OOP-S02

> Java 17 float uses IEEE 754 binary32, and double uses binary64.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean
> and other types).
> • All non-primitive types are reference
> types, so all class types are reference
> types.
> • All integers in Java are signed, Java
> doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 14

Question 43 | Q00445

Which of the following data type(s) can store 64 bit Value.

A. boolean

B. int

C. float

D. long

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer D: long is the 64-bit integer primitive.

OOP06 identifies long as the 64-bit signed integer primitive, following byte, short and int at 8,16 and 32 bits. That directly matches the question's storage-width interpretation. boolean represents logical values, int has only 32 bits, and float is not the stated 64-bit integer storage type.

Why each choice is right or wrong

A - Incorrect: boolean represents logical values rather than providing storage for an arbitrary 64-bit numeric value.

B - Incorrect: The note assigns int a 32-bit width, so it is not the 64-bit integer type.

C - Incorrect: float is a floating numeric type rather than the long integer type explicitly assigned 64 bits.

D - Correct: long is the 64-bit signed integer type identified here's width sequence.

Primary note proof

Study-note section OOP06

> Java has byte, short, int, long, float, double, char and boolean. Signed integer widths are 8,16,32,64 bits respectively.

Study-note section OOP-S02

> Java 17 float uses IEEE 754 binary32, and double uses binary64.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,

and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 15

Question 44 | Q00446

Short data type has a minimum value of _____.

A. -32768

B. -32767

C. 32768

D. 32767

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer A: -32768.

OOP06 gives the Java short range as -32768 through 32767. The question asks for its minimum, so the negative lower endpoint is required. -32767 is one above the minimum; positive 32767 is the maximum, and positive 32768 is outside the stated range entirely.

Why each choice is right or wrong

A - Correct: -32768 is exactly the lower endpoint of the short range given in OOP06.

B - Incorrect: -32767 is representable but is one above the minimum short value.

C - Incorrect: 32768 is above the maximum short value and cannot be its minimum.

D - Incorrect: 32767 is the maximum short value, whereas the question asks for the minimum.

Primary note proof

Study-note section OOP06

byte ranges from -128 to 127; short from -32768 to 32767.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java
(byte, short, int, long, float, double,
char & Boolean)
2. Reference types: Arrays, classes,
and interfaces, String)
• Note: Boolean type has a value true or
false (there is no conversion b/n Boolean
and other types).
• All non-primitive types are reference
types, so all class types are reference
types.
• All integers in Java are signed, Java
doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 16

Question 45 | Q00447

What default value does a Java boolean instance field receive before any explicit initializer or constructor assignment?

A. TRUE

B. false

C. null

D. garbage

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer B: false

Java fields receive defined initial values before ordinary explicit initialization. For primitive boolean fields that value is false. true is the other Boolean value, null is a reference value rather than a Boolean value, and Java does not expose an uninitialized field as garbage. The question specifically selects an instance field, so local-variable definite-assignment rules do not change the answer.

Why each choice is right or wrong

A - Incorrect: true is not the default Boolean field value.

B - Correct: false is the default value of a boolean field.

C - Incorrect: null is a reference default and cannot be a primitive boolean value.

D - Incorrect: Java supplies a defined field default rather than a garbage value.

Primary note proof

Study-note section OOP05

> Java fields and array elements receive defaults: numeric zero, char '\u0000', boolean false and references null. A local int t; followed by printing t is a compilation error.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 5 | lines 14-38

> B. Instance variables:
> • Are declared in a class, but
> outside a method, constructor or
> any block.
> • Have default values.
> • are variables which are bound to
> the object itself. Meaning=>
> • Every instance of that class
> (object) has it's own copy of that
> variable
> • Are created when an object is
> created with the use of the
> keyword 'new' and destroyed
> when the object is destroyed.
> • Can be declared in class level
> before or after use.
> • Access modifiers can be given
> for instance variables.
> • Are visible for all methods,
> constructors and block in the
> class.
> E.g. class MyClass {
> float aFloat; }
> NB: By default, a member declared
> within a class is an instance member.

Object Oriented Programming module | PDF page 4 | lines 59-73

> There are three kinds of variables in Java:
> A. Local variables:
> • Local variables are declared in
> methods, constructors, or blocks.
> • has only local scope
> • Access modifiers cannot be used

for local variables.
• They are visible only within the declared method, constructor, or block.
• Have not a default value
• Should be declared and initialized @ z first time.
• Declare before use anywhere in a method or block.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 17

Question 46 | Q00448

What will be the output of the program?

```
class Main {
public static void main(String args[]) {
int t;
System.out.println(t);
}
}
```

A. 0

B. garbage value

C. compiler error

D. runtime error

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer C: compiler error

t is a local variable read before assignment. Local variables must be initialized before use and have no default value.

Why each choice is right or wrong

A - Incorrect: Incorrectly applies a field-style default

B - Incorrect: Locals cannot be read as an arbitrary garbage value

C - Correct: Violates the initialization-before-use requirement

D - Incorrect: The violation arises before execution rather than as a runtime failure

Primary note proof

Study-note section OOP05

> Locals occur in methods, constructors or blocks; they must be definitely assigned before use and cannot have public/private/protected modifiers.

Study-note section OOP05

> A local int t; followed by printing t is a compilation error.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 59-73

> There are three kinds of variables in Java:
> A. Local variables:
> • Local variables are declared in
> methods, constructors, or blocks.
> • has only local scope
> • Access modifiers cannot be used
> for local variables.
> • They are visible only within the
> declared method, constructor, or
> block.
> • Have not a default value
> • Should be declared and
> initialized @ z first time.
> • Declare before use anywhere in a
> method or block.

Question sources

Object-Oriented Programming practice questions · Question 18

Question 47 | Q00449

What will be the output of the program?
class Test {
public static void main(String[] args) {
for(int i = 0; 0; i++)
{
System.out.println(""Hello"");
break;
}
}
}

A. Hello

B. Empty Output

C. Compiler error

D. Runtime error

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer C: Compiler error.

The for condition is integer literal 0, whereas OOP06 explicitly says for(...;0;...) is invalid in Java. The doubled quotes around Hello are another source syntax defect. Compilation therefore fails before any loop or print can execute. Treating zero as a false Java condition would incorrectly import C++ numeric truth conversion.

Why each choice is right or wrong

A - Incorrect: No Hello output occurs because the invalid loop condition prevents compilation.

B - Incorrect: Zero is not a legal false condition in Java; this is a compile-time error rather than a valid zero-iteration loop.

C - Correct: The numeric condition and malformed quoted text are compile-time source defects.

D - Incorrect: The program fails translation before execution, so the relevant failure is not a runtime exception.

Primary note proof

Study-note section OOP06

> boolean does not convert to or from numeric truth values; if(0) and for(...;0;...) are invalid.

Study-note section OOP-S01

> An identifier is a programmer-chosen name for an entity such as a class, method or variable. A reserved keyword has a language-defined role and cannot be used as an ordinary identifier. A

literal writes a value directly, such as an integer, floating-point number, quoted character, quoted string or Boolean value.

Study-note section OOP-S03

A while header is while(condition). A basic for header is for(initialization; condition; update). A do-while statement places its condition after the body. An if selects its true branch or, when present, its else branch; only the selected branch executes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Object Oriented Programming module | PDF page 4 | lines 11-20

✓ Reserved keywords: class, extends, implements, for…
✓ Identifiers: programmers given name; grade, gRade…
✓ Literals (Constants values): 55, 60.78, “Hi”, „M‟… Java language specifies five major types of literals => integer, floating, character, string & Boolean literals

Question sources

Object-Oriented Programming practice questions · Question 19

Question 48 | Q00450

What will be the output of the program?

```
class mainclass {
public static void main(String args[])
{
boolean var1 = true;
boolean var2 = false;
if (var1)
System.out.println(var1);
else
System.out.println(var2);
}
}
```

A. 0

B. 1

C. true

D. FALSE

Hint

Respect operator grouping and operand types, then trace each operation in evaluation order.

Answer C: true

var1 is initialized to true, so the if body executes and prints var1. The else body is skipped, so var2 does not appear. Java's Boolean output is the lowercase word true, followed by the newline from println. It does not use numeric 1 or uppercase TRUE. Correcting C's case makes it the exact visible output rather than an approximate description.

Why each choice is right or wrong

A - Incorrect: The chosen true Boolean value does not print as numeric 0.

B - Incorrect: Java Boolean output is not encoded as numeric 1.

C - Correct: The true branch prints the lowercase word true.

D - Incorrect: The false value is not printed because the else branch is skipped.

Primary note proof

Study-note section OOP23

> Printing Boolean true gives lowercase true, not 1 or uppercase TRUE.

Study-note section OOP-S03

> A while header is while(condition). A basic for header is for(initialization; condition; update). A do-while statement places its condition after the body. An if selects its true branch or, when present, its else branch; only the selected branch executes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Object Oriented Programming module | PDF page 7 | lines 48-54

2. Declaration statements: declares/ Introduces a very first variables E.g. double c = 8.4;
3. Control flow statements: regulates the order of statements execution
Selection: the if() and switch() statements

Question sources

Object-Oriented Programming practice questions · Question 20

Question 49 | Q00451

What is the numerical value computed by the final expression, rounded to eight digits after the decimal point? This asks for the rounded value, not the exact unformatted console string.

```
class average {
public static void main(String args[])
{
double num[] = {5.5, 10.1, 11, 12.8, 56.9, 2.5};
double result;
result = 0;
for (int i = 0; i<6; ++i)
result = result + num[i];
System.out.print(result/6);
}
}
```

A. 16.34

B. 16.56666664

C. 16.46666667

D. 16.76666667

Hint

Trace every element or API argument and distinguish copying values from aliasing a reference.

Answer C: 16.46666667

The six listed numbers sum to 98.8 mathematically. Dividing by six gives approximately 16.4666666667, which rounds to 16.46666667 at eight decimal places. Small binary-double representation differences do not affect that rounding here. The original unformatted print call does not impose eight-place formatting, so the revised question explicitly requests a rounded numerical value rather than claiming this is its exact console string.

Why each choice is right or wrong

A - Incorrect: 16.34 is not the sum divided by six at the requested precision.

B - Incorrect: 16.56666664 differs from the computed mean before rounding.

C - Correct: 16.46666667 is the mean rounded to eight decimal places.

D - Incorrect: 16.76666667 does not match the computed mean.

Primary note proof

Study-note section OOP27

> For int[] a=new int[4], setting a[i]=i*3 and accumulating yields 0+3+6+9=18. To average an array, sum every element and divide by its length using floating arithmetic; handle zero length separately. Floating representations and printing can differ from a rounded mathematical result. For the six-value example, the mathematical average is 98.8/6, approximately 16.46666667; the exact printed string also depends on floating representation and formatting.

Study-note section OOP26

Valid indices are 0..a.length-1; an invalid access throws ArrayIndexOutOfBoundsException.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 21

Question 50 | Q00452

What will be the output of the program?
class increment {
public static void main(String args[])
{
int g = 4;
System.out.print(++g * 8);
}
}

A. 32

B. 36

C. 40

D. 48

Hint

Track the old and updated values separately and distinguish a character operand from a string.

Answer C: 40.

The prefix increment changes g from 4 to 5 before its value is used in multiplication. The expression therefore computes 5*8=40 and prints that integer. This differs from postfix increment, which would supply the old value to the multiplication. Only one increment occurs in this expression.

Why each choice is right or wrong

A - Incorrect: 32 would multiply the old value 4 by 8, which does not respect prefix increment.

B - Incorrect: 36 is not the product of either the old value 4 or updated value 5 with 8.

C - Correct: Prefix increment supplies 5, so the printed product is 5 times 8, or 40.

D - Incorrect: 48 requires g to become 6, but the single increment changes it only from 4 to 5.

Primary note proof

Study-note section OOP24

> Starting x=5, printing x++ then ++x gives 5 then 7.

Study-note section OOP23

> Java evaluates operands left to right while respecting precedence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that
> take one or more arguments

(operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Question sources

Object-Oriented Programming practice questions · Question 22

Question 51 | Q00453

What will this program print when run on Java 17?

```
class area {
public static void main(String args[])
{
double r, pi, a;
r = 9.8;
pi = 3.14;
a = pi * r * r;
System.out.println(a);
}
}
```

A. 301.5656

B. 301

C. 301.56

D. 301.57

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer A: 301.5656

Java groups pi * r * r as (pi * r) * r and rounds each double multiplication using binary64 arithmetic. The resulting value has hexadecimal representation 0x1.2d90cb295e9e2p+8. Java 17's standard conversion prints that value as 301.5656, followed by println's line terminator. The supplementary Oracle rules establish the precise arithmetic and text conversion; no separate decimal-place formatting is applied.

Why each choice is right or wrong

A - Correct: 301.5656 is the standard Java 17 printed representation of the computed double.

B - Incorrect: There is no integer conversion that discards the fractional part.

C - Incorrect: The program does not truncate or format the result to two decimal places.

D - Incorrect: The program does not round its displayed output to two decimal places.

Primary note proof

Study-note section OOP-S02

> Java 17 float uses IEEE 754 binary32, and double uses binary64. Successive multiplications group from left to right. Each floating-point operation rounds to its representable result, using nearest-value rounding with ties to even; regrouping a product can change its result.

Study-note section OOP-S02

> PrintStream.print(double) converts the stored value through String.valueOf and Double.toString. For finite nonzero magnitudes from 0.001 inclusive to 10000000 exclusive, the standard decimal form uses at least one fractional digit and only the further digits needed to distinguish the value from adjacent doubles. println(double) also adds a line terminator. This is not fixed-decimal-place formatting.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 23

Question 52 | Q00454

What will be the output of the program?

```
class increment {
public static void main(String args[])
{
int g = 6;
System.out.print(--g * 8);
}
}
```

A. 48

B. 40

C. 56

D. 44

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer B: 40.

The prefix decrement reduces g from 6 to 5 before multiplication uses it. The resulting integer expression is 5*8, which equals 40. The updated value is used immediately; it is not the old 6 and no later increment occurs. Thus the integer printed is 40.

Why each choice is right or wrong

A - Incorrect: 48 uses the old value 6, which would not reflect the prefix decrement before multiplication.

B - Correct: Prefix decrement supplies 5 to the multiplication, producing 5*8=40.

C - Incorrect: 56 uses 7, but decrement reduces the initial 6 instead of increasing it.

D - Incorrect: 44 is not 5 multiplied by 8 and does not follow the stated prefix update.

Primary note proof

Study-note section OOP-S03

> Prefix ++ and -- update a variable before supplying its value; postfix forms supply the old value before the update. Multiplication, division and remainder bind before addition and subtraction; same-level binary arithmetic groups left to right. Integer division truncates toward zero.

Study-note section OOP23

> Java evaluates operands left to right while respecting precedence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Question sources

Object-Oriented Programming practice questions · Question 24

Question 53 | Q00456

To declare more than one variable of the specified type, we can use a _____ list.

A. colon-separated

B. bracket-separated

C. comma-separated

D. None of the above

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer C: A comma-separated declarator list.

OOP05 illustrates a declaration with int a=10,b=10, separating the declarators with a comma. The shared type applies to each name in that declaration. Colons and brackets serve other syntactic purposes and do not separate ordinary variable declarators in the example. The comma-separated option states the required list form.

Why each choice is right or wrong

A - Incorrect: A colon does not separate the ordinary variable declarators shown in the note's shared-type declaration.

B - Incorrect: Brackets describe array types or indexing rather than separating names in a declarator list.

C - Correct: The example int a=10,b=10 uses a comma between variable declarators sharing the type int.

D - Incorrect: Comma-separated is an offered and supported list form, so none of the above is false.

Primary note proof

Study-note section OOP05

> Initializers such as int a=10,b=10; assign initial values.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 11-20

> ✓ Reserved keywords: class, extends, implements, for…
> ✓ Identifiers: programmers given name; grade, gRade…
> ✓ Literals (Constants values): 55, 60.78, "Hi", „M"… Java language specifies five major types of literals => integer,

floating, character, string &
Boolean literals

Question sources

Object-Oriented Programming practice questions · Question 26

Question 54 | Q00457

In the introductory grouping by declaration scope and class-versus-instance ownership, how many categories of Java variables are distinguished? Include parameter and block-local contexts within the local category.

A. 2

B. 3

C. 4

D. 5

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer B: 3

The introductory grouping distinguishes local variables, instance fields and class/static fields. Locals belong to method, constructor or block execution; instance fields belong to individual objects; static fields belong to the class. This gives three categories. The question uses that teaching grouping rather than counting every variable category distinguished by the full language specification.

Why each choice is right or wrong

A - Incorrect: Two categories cannot account for all three of local, instance and class/static variables.

B - Correct: The specified introductory grouping contains local, instance and class/static categories: three.

C - Incorrect: Four exceeds the three categories in the specified grouping.

D - Incorrect: Five exceeds the three categories in the specified grouping.

Primary note proof

Study-note section OOP05

The introductory three-way grouping is local, instance and class/static variables.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 59-73

There are three kinds of variables in Java:
A. Local variables:
• Local variables are declared in methods, constructors, or blocks.
• has only local scope
• Access modifiers cannot be used for local variables.
• They are visible only within the declared method, constructor, or block.
• Have not a default value
• Should be declared and initialized @ z first time.
• Declare before use anywhere in a method or block.

Object Oriented Programming module | PDF page 5 | lines 14-38

B. Instance variables:
• Are declared in a class, but outside a method, constructor or any block.
• Have default values.
• are variables which are bound to the object itself. Meaning=>
• Every instance of that class (object) has it's own copy of that variable
• Are created when an object is created with the use of the keyword 'new' and destroyed when the object is destroyed.
• Can be declared in class level before or after use.
• Access modifiers can be given for instance variables.
• Are visible for all methods, constructors and block in the class.
E.g. class MyClass {
float aFloat; }
NB: By default, a member declared within a class is an instance member.

Object Oriented Programming module | PDF page 5 | lines 39-61

C. Class/Static variables
• Are declared with the static

keyword in a class, but outside a method, constructor or a block.
• There would only be one copy of each class variable per class, regardless of how many objects are created from it.
• Are rarely used other than being declared as constants. (public static final)
• Are stored in the static memory.
• Static variables are created when the program starts and destroyed when the program stops.
• Visibility is similar to instance variables. However, most static variables are declared public
• Default values are same as instance variables.
• Static variables can be accessed by calling with the class name ClassName.VariableName.

Question sources

Object-Oriented Programming practice questions · Question 27

Question 55 | Q00458

Local variables are declared in?

A. methods

B. constructors

C. blocks

D. All of the above

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer D: All of the above

The local-variable definition explicitly includes methods, constructors and blocks.

Why each choice is right or wrong

A - Incorrect: One of the three listed contexts

B - Incorrect: One of the three listed contexts

C - Incorrect: One of the three listed contexts

D - Correct: Combines all three permitted contexts

Primary note proof

Study-note section OOP05

> Locals occur in methods, constructors or blocks; they must be definitely assigned before use and cannot have public/private/protected modifiers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 59-73

> There are three kinds of variables in Java:
> A. Local variables:
> • Local variables are declared in methods, constructors, or blocks.
> • has only local scope
> • Access modifiers cannot be used for local variables.
> • They are visible only within the declared method, constructor, or block.
> • Have not a default value
> • Should be declared and initialized @ z first time.
> • Declare before use anywhere in a method or block.

Question sources

Object-Oriented Programming practice questions · Question 28

Question 56 | Q00459

Which qualifier is preferred for accessing an accessible Java static field to make its class-level ownership clear?

A. Object name

B. Class name

C. Function name

D. Can not say

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer B: Class name

A static field belongs to the loaded class and is shared by its instances. ClassName.field makes that ownership explicit and is the recommended qualification. An object reference can sometimes qualify a static member, but it obscures the class-level nature and is not the preferred form asked for. Function names do not supply the owning type, and the style rule gives a definite answer.

Why each choice is right or wrong

A - Incorrect: Object qualification can be legal, but it is not the preferred class-ownership notation.

B - Correct: Class qualification is the preferred notation for a static field.

C - Incorrect: A function name does not identify the field's owning type as a qualifier.

D - Incorrect: The note provides a definite preferred class-qualified form.

Primary note proof

Study-note section OOP07

> static gives one field per loaded class, shared by its instances. If count starts at zero and every constructor increments it, three creations leave count=3. Prefer ClassName.count and ClassName.utility().

Study-note section OOP16

> With Animal a=new Dog(), a.staticMethod() resolves using Animal's compile-time type. Prefer Animal.staticMethod() for clarity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 5 | lines 39-61

> C. Class/Static variables
> • Are declared with the static keyword in a class, but outside a method, constructor or a block.
> • There would only be one copy of each class variable per class, regardless of how many objects are created from it.
> • Are rarely used other than being declared as constants. (public static final)
> • Are stored in the static memory.
> • Static variables are created when the program starts and destroyed when the program stops.
> • Visibility is similar to instance variables. However, most static

variables are declared public
• Default values are same as instance variables.
• Static variables can be accessed by calling with the class name ClassName.VariableName.

Object Oriented Programming module | PDF page 14 | lines 2-19

Static methods and Final Variables
• A static method can access only static data.
• A static method can call only other static data (Static methods & variables), but can not call a non-static data from it.
• Static method is belongs to the class and not to the object.
• A static method can be accessed directly by the class name and doesn‟t need any object.
• i.e ClassName.methodName(args)
• A static method cannot refer to "this" or "super" keywords in anyway
• But can be overloaded since they are resolved using static binding by compiler at compile time.

Question sources

Object-Oriented Programming practice questions · Question 29

Question 57 | Q00460

Which of the following is an Example of variable initialization?

A. int a, b, c;

B. int a = 10, b = 10;

C. int 10 = a;

D. None of the above

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer B: int a = 10, b = 10;

The declaration double c = 8.4 illustrates assignment of an initial value at declaration; B follows the same type-name-equals-value pattern for a and b.

Why each choice is right or wrong

A - Incorrect: Declares variables without initial values

B - Correct: Assigns initial values 10 to named variables

C - Incorrect: Places a literal where the declared name belongs

D - Incorrect: B demonstrates initialization

Primary note proof

Study-note section OOP05

> Initializers such as int a=10,b=10; assign initial values.

Study-note section OOP-S01

> An identifier is a programmer-chosen name for an entity such as a class, method or variable. A reserved keyword has a language-defined role and cannot be used as an ordinary identifier. A literal writes a value directly, such as an integer, floating-point number, quoted character, quoted string or Boolean value.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 7 | lines 48-54

> 2. Declaration statements: declares/
> Introduces a very first variables E.g.
> double c = 8.4;
> 3. Control flow statements: regulates
> the order of statements execution
> Selection: the if() and switch()
> statements

Question sources

Object-Oriented Programming practice questions · Question 30

Question 58 | Q00461

Access modifiers cannot be used for local variables.

A. Yes

B. No

C. Can be yes or no

D. Cannot say

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer A: Yes

The local-variable section explicitly says access modifiers cannot be used for local variables.

Why each choice is right or wrong

A - Correct: Agrees with the rule

B - Incorrect: Contradicts the rule

C - Incorrect: The stated prohibition is not conditional

D - Incorrect: The rule gives a definite answer

Primary note proof

Study-note section OOP05

> Locals occur in methods, constructors or blocks; they must be definitely assigned before use and cannot have public/private/protected modifiers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 59-73

> There are three kinds of variables in Java:
> A. Local variables:
> • Local variables are declared in
> methods, constructors, or blocks.
> • has only local scope
> • Access modifiers cannot be used
> for local variables.
> • They are visible only within the
> declared method, constructor, or
> block.
> • Have not a default value
> • Should be declared and
> initialized @ z first time.
> • Declare before use anywhere in a
> method or block.

Question sources

Object-Oriented Programming practice questions · Question 31

Question 59 | Q00462

Which of the following can be operands of arithmetic operators?

A. Characters

B. Boolean

C. Numeric

D. Both Numeric & Characters

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer D: Both numeric values and characters can participate in arithmetic.

OOP06 states that arithmetic promotes char, byte and short to int in many expressions. Characters are therefore allowed alongside numeric operands. boolean does not convert to numeric truth values and is not an arithmetic operand. D gives the complete supported pair; A and C each mention only part of that operand domain.

Why each choice is right or wrong

A - Incorrect: Characters are permitted through promotion, but this option alone omits the also-permitted numeric operand category.

B - Incorrect: Java boolean does not convert to numeric values for arithmetic, as OOP06 explicitly states.

C - Incorrect: Numeric operands are permitted, but this option alone omits the also-permitted character category.

D - Correct: Both numeric and char operands are permitted; chars undergo numeric promotion for arithmetic.

Primary note proof

Study-note section OOP06

> Arithmetic promotes char, byte and short to int in many expressions. boolean does not convert to or from numeric truth values; if(0) and for(...;0;...) are invalid.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary),

or three operands (Ternary).
Java has one ternary operator,
?:, which is a short-hand if-
else statement. E.g. c=x?w:s;
Operators can be Arithmetic,
Logical, assignment,
Relational, conditional,
Increment,

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java
(byte, short, int, long, float, double,
char & Boolean)
2. Reference types: Arrays, classes,
and interfaces, String)
• Note: Boolean type has a value true or
false (there is no conversion b/n Boolean
and other types).
• All non-primitive types are reference
types, so all class types are reference
types.
• All integers in Java are signed, Java
doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 32

Question 60 | Q00463

Modulus operator, %, can be applied to which of these?

A. Both Integers and floating - point numbers

B. Integers

C. Floating - point numbers

D. None of the mentioned

Hint

Respect operator grouping and operand types, then trace each operation in evaluation order.

Answer A: Both integers and floating-point numbers in Java.

OOP23 explicitly contrasts Java with C++: Java % accepts floating-point operands as well as integers. This course's Java context therefore makes A the complete domain among the offered choices. B and C each name only one supported category. Importing C++'s integral-only restriction would give the wrong answer for Java.

Why each choice is right or wrong

A - Correct: Both integer and floating-point operands are accepted by Java's remainder operator according to OOP23.

B - Incorrect: Integers are allowed, but this option alone omits Java's also-supported floating-point operands.

C - Incorrect: Floating-point values are allowed, but this option alone omits the also-supported integer operands.

D - Incorrect: The note explicitly supports both supplied operand categories, so none of the mentioned is false.

Primary note proof

Study-note section OOP23

> Unlike C++, Java % also accepts floating-point operands.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Question sources

Object-Oriented Programming practice questions · Question 33

Question 61 | Q00465

Evaluate the value of the expression?
6 - 2 + 10 % 4 + 7

A. 14

B. 12

C. 13

D. 10

Hint

Respect operator grouping and operand types, then trace each operation in evaluation order.

Answer C: 13.

The remainder operation is evaluated first: 10%4 is 2. The remaining additions and subtraction group left to right, yielding 6-2+2+7 = 4+2+7 = 13. OOP23 includes the same expression and result. No floating conversion or increment operation appears in this calculation.

Why each choice is right or wrong

A - Incorrect: 14 does not follow the remainder 10%4=2 and the resulting sum 6-2+2+7.

B - Incorrect: 12 is one below the computed total and does not respect the stated remainder result.

C - Correct: Evaluating 10%4 first gives 2; then 6-2+2+7 equals 13.

D - Incorrect: 10 omits or miscalculates part of the expression; the full integer evaluation gives 13.

Primary note proof

Study-note section OOP23

> Java evaluates operands left to right while respecting precedence.

Study-note section OOP23

> 6-2+10%4+7 is 13.

Study-note section OOP-S03

> Prefix ++ and -- update a variable before supplying its value; postfix forms supply the old value before the update. Multiplication, division and remainder bind before addition and subtraction; same-level binary arithmetic groups left to right. Integer division truncates toward zero.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that
> take one or more arguments
> (operands) and operates on
> them to produce a result. An

operator performs a function
on one (Unary), two (Binary),
or three operands (Ternary).
Java has one ternary operator,
?:, which is a short-hand if-
else statement. E.g. c=x?w:s;
Operators can be Arithmetic,
Logical, assignment,
Relational, conditional,
Increment,

Question sources

Object-Oriented Programming practice questions · Question 35

Question 62 | Q00466

What is the output of this program?

```
class Main {
public static void main(String args[])
{
double var1 = 2 + 4;
double var2 = var1 / 4;
int var3 = 2 + 4;
int var4 = var3 / 4;
System.out.print(var2 + " " + var4);
}
}
```

A. 0 1

B. 1 1

C. 1.5 1

D. 1.5 1.0

Hint

Respect operator grouping and operand types, then trace each operation in evaluation order.

Answer C: 1.5 1.

Both additions initially produce 6. Assigning one to double makes var1=6.0, so var1/4 gives 1.5. var3 remains int, so var3/4 truncates to integer 1. The string containing a space then concatenates their representations, producing 1.5 1 rather than giving the integer result a decimal suffix.

Why each choice is right or wrong

A - Incorrect: The double division of 6.0 by 4 yields 1.5, not zero.

B - Incorrect: The first quotient is floating point and retains .5 rather than becoming integer 1.

C - Correct: The double quotient prints 1.5, followed by the space and integer quotient 1.

D - Incorrect: var4 is int, so its string representation is 1 rather than 1.0.

Primary note proof

Study-note section OOP23

> Integer 6/4 is 1; double 6.0/4 is 1.5.

Study-note section OOP24

> Once a String operand is encountered, + concatenates in left-to-right groups: 1+2+"x" gives "3x", while "x"+1+2 gives "x12".

Study-note section OOP-S03

> A while header is while(condition). A basic for header is for(initialization; condition; update). A do-while statement places its condition after the body. An if selects its true branch or, when present, its else branch; only the selected branch executes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that
> take one or more arguments
> (operands) and operates on
> them to produce a result. An
> operator performs a function
> on one (Unary), two (Binary),
> or three operands (Ternary).
> Java has one ternary operator,
> ?:, which is a short-hand if-
> else statement. E.g. c=x?w:s;
> Operators can be Arithmetic,
> Logical, assignment,
> Relational, conditional,
> Increment,

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean
> and other types).
> • All non-primitive types are reference

types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 36

Question 63 | Q00467

What will be the output of the program?

```
class Bitwise
{
public static void main(String [] args)
{
int x = 11 & 9;
int y = x ^ 3;
System.out.println( y | 12 );
}
}
```

A. 7

B. 0

C. 14

D. 8

Hint

Respect operator grouping and operand types, then trace each operation in evaluation order.

Answer C: 14.

The AND produces 11&9 = 1011&1001 = 1001, or 9. XOR with 3 gives 1001^0011 = 1010, or 10. OR with 12 then gives 1010|1100 = 1110, or 14. OOP23 explicitly provides this chain; these are integer bitwise operations, not Boolean short-circuit operations.

Why each choice is right or wrong

A - Incorrect: Seven has binary 0111, whereas the final OR produces 1110.

B - Incorrect: The final OR includes the set bits of 12, so it cannot produce zero.

C - Correct: The chain produces x=9, y=10 and final y|12=14.

D - Incorrect: Eight omits bits set by the final OR of binary 1010 and 1100.

Primary note proof

Study-note section OOP23

For bitwise values, write binary columns: 11&9=9; 9^3=10; 10|12=14.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Question sources

Object-Oriented Programming practice questions · Question 37

Question 64 | Q00468

What is the output of relational operators?

A. Integer

B. Boolean

C. Characters

D. Double

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer B: Boolean.

Relational expressions test a relation rather than return one of their numeric operands. In Java that truth result is Boolean, consistent with OOP06's prohibition on treating boolean as numeric truth values and OOP23's Boolean output discussion. Integer, character and double values can be compared in suitable expressions, but are not the comparison result type.

Why each choice is right or wrong

A - Incorrect: Java comparison results are logical values rather than integer truth codes such as zero and one.

B - Correct: A relational test yields the Boolean truth value true or false.

C - Incorrect: Characters can be compared, but a comparison does not return a character result.

D - Incorrect: Double operands may participate in comparisons, but their relation yields a Boolean rather than a double.

Primary note proof

Study-note section OOP-S03

> Java relational comparisons (<, <=, >, >=) and equality tests (==, !=) produce primitive boolean truth values. Their result is not an integer truth code or a value of the numeric operand type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
> 2. Reference types: Arrays, classes, and interfaces, String)
> • Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
> • All non-primitive types are reference types, so all class types are reference types.

• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 38

Question 65 | Q00469

Which of these is returned by "greater than", "less than" and "equal to" operators?

A. Integers

B. Floating - point numbers

C. Boolean

D. None of the mentioned

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer C: Boolean true or false.

Greater-than, less-than and equality operators express truth tests. Java represents their results as Boolean values, not integers or floating-point numbers. OOP06 distinguishes boolean from numeric values, and OOP23 describes printing Boolean true as lowercase true. The operand type therefore does not become the result type of the test.

Why each choice is right or wrong

A - Incorrect: Java uses Boolean truth values for comparisons rather than returning integer zero or one.

B - Incorrect: Comparing floating operands still yields a truth value, not a floating-point numerical result.

C - Correct: Greater-than, less-than and equality comparisons return Boolean true or false.

D - Incorrect: Boolean is explicitly offered and is the appropriate logical result type, so none is false.

Primary note proof

Study-note section OOP-S03

Java relational comparisons (<, <=, >, >=) and equality tests (==, !=) produce primitive boolean truth values. Their result is not an integer truth code or a value of the numeric operand type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 39

Question 66 | Q00471

In Java, what type must expression1 have in the conditional expression expression1 ? expression2 : expression3?

A. Integer

B. Floating - point numbers

C. Boolean

D. None of the mentioned

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer C: Java requires boolean or Boolean for the first operand of ?: .

Java's conditional operator requires its first operand to have primitive boolean or boxed Boolean type. That condition chooses which of the other two expressions is evaluated; the unselected alternative is not evaluated. Integer and floating-point values are not accepted as numeric truth codes. The Boolean option therefore names the required logical category, and None is false because that category is offered.

Why each choice is right or wrong

A - Incorrect: Java integer values do not satisfy the JLS first-operand requirement of boolean or Boolean.

B - Incorrect: Floating-point values do not satisfy Java's Boolean conditional-operand requirement.

C - Correct: Boolean names the required logical category; Java accepts primitive boolean or boxed Boolean.

D - Incorrect: The Boolean category is offered and satisfies the JLS requirement, so none is false.

Primary note proof

Study-note section OOP-S03

> The conditional expression condition ? whenTrue : whenFalse requires a boolean or Boolean condition. It evaluates only the selected alternative. An integer or floating-point first operand is a compile-time error.

Study-note section OOP06

> boolean does not convert to or from numeric truth values; if(0) and for(...;0;...) are invalid.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic,

Logical, assignment,
Relational, conditional,
Increment,

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java
(byte, short, int, long, float, double,
char & Boolean)
2. Reference types: Arrays, classes,
and interfaces, String)
• Note: Boolean type has a value true or
false (there is no conversion b/n Boolean
and other types).
• All non-primitive types are reference
types, so all class types are reference
types.
• All integers in Java are signed, Java
doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 41

Question 67 | Q00472

What is the value stored in x in following lines of code?
int x, y, z;
x = 0;
y = 1;
x = y = z = 8;

A. 0

B. 1

C. 9

D. 8

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer D: x becomes 8.

The last statement assigns 8 to z, then that same assigned value to y and then x. The earlier x=0 and y=1 values are overwritten, and no addition takes place. This follows assignment's value propagation rather than the left-to-right operand arithmetic discussed in OOP23. The final stored value of x is 8.

Why each choice is right or wrong

A - Incorrect: Zero is x's earlier value; the final chained assignment overwrites it with 8.

B - Incorrect: One is y's earlier value, which is also overwritten before x receives the chain's result.

C - Incorrect: The statement performs assignments, not addition of the earlier y value to 8.

D - Correct: The chain assigns 8 to z, then y and x, leaving x equal to 8.

Primary note proof

Study-note section OOP-S03

> Assignments group right to left: a=b=c assigns c to b, then the resulting value to a. A later assignment overwrites an earlier value. Compound assignments update the left variable using the indicated operation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Question sources

Object-Oriented Programming practice questions · Question 42

Question 68 | Q00473

What is the output of this program?

```
class Main
{
public static void main(String args[])
{
int x = 8;
System.out.println(++x * 3 + " " + x);
}
}
```

A. 24 8

B. 24 9

C. 27 8

D. 27 9

Hint

Track the old and updated values separately and distinguish a character operand from a string.

Answer D: 27 9.

Prefix ++x changes x from 8 to 9 before multiplication, producing 27. Concatenating the string containing a space switches the remaining + operations to text concatenation. The later x still holds 9 because no further update occurs. Therefore the single println emits 27 9, followed by its newline.

Why each choice is right or wrong

A - Incorrect: Both values wrongly use the original x=8 despite the prefix increment before multiplication.

B - Incorrect: The final x is 9, but the product must also use updated 9 and become 27.

C - Incorrect: The product is 27, but x remains 9 rather than reverting to its original 8.

D - Correct: The prefix update gives 9, its product is 27, and concatenation then appends space and 9.

Primary note proof

Study-note section OOP24

> Starting x=5, printing x++ then ++x gives 5 then 7.

Study-note section OOP24

> Once a String operand is encountered, + concatenates in left-to-right groups: 1+2+"x" gives "3x", while "x"+1+2 gives "x12".

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Question sources

Object-Oriented Programming practice questions · Question 43

Question 69 | Q00474

_____ are a sequence of characters.

A. Character

B. Strings

C. Integer

D. Classes

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer B: Strings

The String Class description explicitly equates strings with a sequence of chars.

Why each choice is right or wrong

A - Incorrect: A character is one element, not the named sequence type

B - Correct: The sequence-of-characters type

C - Incorrect: Integer represents a different primitive category

D - Incorrect: Class is the general blueprint concept rather than this sequence type

Primary note proof

Study-note section OOP-S04

> A String represents a sequence of UTF-16 code units. A single char represents one code unit rather than an entire String. String objects are immutable; changing a reference to another String does not modify the original text value.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 14-16

> 5. String Class:- Immutable
> Class created to manipulate
> strings(Sequence of chars).

Question sources

Object-Oriented Programming practice questions · Question 44

Question 70 | Q00475

The String class is?

A. mutable

B. immutable

C. Both A and B

D. None of the above

Hint

Separate identity, content equality, immutable values and references that may be reassigned.

Answer B: immutable

String is immutable, unlike mutable classes whose state can change.

Why each choice is right or wrong

A - Incorrect: Contradicts the stated immutable classification

B - Correct: Matches the stated classification

C - Incorrect: The two opposite classifications do not both describe String here

D - Incorrect: B is directly supported

Primary note proof

Study-note section OOP22

> Java String objects are immutable. text += " World" assigns a reference to a new concatenated result and can print Hello World without mutating the original String object. final class String prevents subclassing; final alone does not make arbitrary objects immutable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 14-16

> 5. String Class:- Immutable
> Class created to manipulate
> strings(Sequence of chars).

Question sources

Object-Oriented Programming practice questions · Question 45

Question 71 | Q00476

Methods used to obtain information about an object are known as?

A. string methods

B. class methods

C. object method

D. accessor methods

Hint

Separate parameters from arguments and operations that read state from operations that change it.

Answer D: Accessor methods.

OOP11 says an accessor or getter returns information, whereas a mutator or setter changes state. Methods used to obtain information about an object therefore have the specific name accessor methods. String, class and object describe other categories or associations and do not identify this information-reading responsibility.

Why each choice is right or wrong

A - Incorrect: String methods concern a particular class; the requested information-reading role applies to objects generally.

B - Incorrect: Class methods identify a class/static association, rather than the specific purpose of returning object information.

C - Incorrect: Object method is too general; object methods can either read or change state.

D - Correct: Accessor methods specifically return information about object state, matching the definition in OOP11.

Primary note proof

Study-note section OOP11

> An accessor/getter returns information; a mutator/setter changes state.

Study-note section OOP07

> Prefer ClassName.count and ClassName.utility().

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 10 | lines 62-72

> Methods
> • Are members of classes and do operations.
> • Allows different classes to communicate each other.

• E.g. type name(parameter-
list)
{ // body of method
//....
return value; // if
type is not void

Object Oriented Programming module | PDF page 10 | lines 15-26

• Every Dog object will contain its own copies of the instance variables breed, age, and color.
• To access these variables, you will use the dot (.) operator.
• The dot operator links the name of the object with the name of an instance variable.
• For example, to assign the age variable of myDog the value 10, you would use the following statement:
myDog.age = 10; //This is called

Question sources

Object-Oriented Programming practice questions · Question 46

Question 72 | Q00477

Which method returns the character at the specified index?

A. int compareTo(Object o)

B. int compareTo(String anotherString)

C. char charAt(int index)

D. int compareToIgnoreCase(String str)

Hint

Separate identity, content equality, immutable values and references that may be reassigned.

Answer C: char charAt(int index).

OOP22 explicitly says charAt(index) returns a char. C gives that operation's parameter and result type: an integer position selects a single character. The compareTo alternatives return int comparison results rather than the character at a position, so they do not satisfy the requested retrieval behavior.

Why each choice is right or wrong

A - Incorrect: This comparison signature takes an object rather than an index and returns int instead of a character.

B - Incorrect: This comparison signature takes another String and returns int, rather than retrieving a character by index.

C - Correct: charAt takes an integer index and returns the char at that position, matching OOP22.

D - Incorrect: This comparison signature takes a String, not an integer index, and does not return the indexed char.

Primary note proof

Study-note section OOP22

> charAt(index) returns a char; length() counts UTF-16 units; substring(begin,end) excludes end.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 14-16

> 5. String Class:- Immutable
> Class created to manipulate
> strings(Sequence of chars).

Question sources

Object-Oriented Programming practice questions · Question 47

Question 73 | Q00478

Which of these methods of String class is used to obtain character at specified index?

A. char()

B. Charat()

C. charat()

D. charAt()

Hint

Separate identity, content equality, immutable values and references that may be reassigned.

Answer D: charAt (with an index argument).

The operation named in OOP22 is charAt(index), with a lowercase c and uppercase A. D alone preserves that exact Java method name. The options abbreviate away the index argument; an actual call needs it. Java's case sensitivity prevents Charat or charat from naming the same standard operation.

Why each choice is right or wrong

A - Incorrect: char is a primitive type keyword, not the indexed String character-access method name.

B - Incorrect: Charat has the wrong capitalization and does not name the charAt operation in OOP22.

C - Incorrect: charat lacks the uppercase A required by the case-sensitive standard method name.

D - Correct: charAt is the exact method name; a real invocation supplies an index inside the parentheses.

Primary note proof

Study-note section OOP22

> charAt(index) returns a char; length() counts UTF-16 units; substring(begin,end) excludes end.

Study-note section OOP28

> Java identifiers are case sensitive; $ is allowed but @ is not an identifier character.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 14-16

> 5. String Class:- Immutable
> Class created to manipulate
> strings(Sequence of chars).

Question sources

Object-Oriented Programming practice questions · Question 48

Question 74 | Q00480

Which Java loop form pre-tests its body condition and writes that condition alone in parentheses immediately after its loop keyword, rather than using a three-clause header or a post-test?

A. for

B. do-while

C. while

D. None of the above

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer C: while

A while loop tests its condition before every possible execution and has a condition-only header. A for loop is also a pre-test loop, but its ordinary header has initialization, condition and update clauses. A do-while loop tests after its body and therefore runs at least once. The added header and timing distinctions make while the unique match rather than relying on repetition alone.

Why each choice is right or wrong

A - Incorrect: A basic for uses the three-clause header excluded by the stem.

B - Incorrect: A do-while post-tests rather than pre-testing its first iteration.

C - Correct: while has the condition-only pre-test form described.

D - Incorrect: The while form satisfies the description, so none is incorrect.

Primary note proof

Study-note section OOP-S03

> A while header is while(condition). A basic for header is for(initialization; condition; update). A do-while statement places its condition after the body. An if selects its true branch or, when present, its else branch; only the selected branch executes.

Study-note section OOP25

> while and for pre-test and may execute zero times; do-while post-tests and executes once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 7 | lines 26-41

> Iteration: The for(), while() and
> do...while() statements
> 4. Jumping Statements: to jump out
> of loops and to control other areas of

program java uses break and continue statements.
The break statement: used to exit a loop
The Continue Statement: used to force program to go back to the top of a loop
Labels: Java does not include a goto statement.
Instead of goto, Java allows you to combine break and continue with a label.

Question sources

Object-Oriented Programming practice questions · Question 50

Question 75 | Q00481

Which loop guaranteed to execute at least one time?

A. for

B. do-while

C. while

D. All of the above

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer B: do-while.

OOP25 states that do-while post-tests and executes once, whereas for and while pre-test and may execute zero times. The mandatory first body execution is therefore specific to do-while among these forms. This guarantee concerns entering its body when the statement is reached, before deciding whether to repeat.

Why each choice is right or wrong

A - Incorrect: A for loop tests its condition before its first body execution and may execute zero times.

B - Correct: A do-while executes its body before testing the condition, guaranteeing the first body execution.

C - Incorrect: A while loop pre-tests; an initially false condition prevents all body executions.

D - Incorrect: For and while can execute zero times, so the first-execution guarantee does not apply to all forms.

Primary note proof

Study-note section OOP25

while and for pre-test and may execute zero times; do-while post-tests and executes once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 7 | lines 26-41

Iteration: The for(), while() and
do...while() statements
4. Jumping Statements: to jump out
of loops and to control other areas of
program java uses break and
continue statements.
The break statement: used to exit a
loop
The Continue Statement: used to
force program to go back to the top
of a loop
Labels: Java does not include a goto
statement.
Instead of goto, Java allows you to
combine break and continue with a
label.

Question sources

Object-Oriented Programming practice questions · Question 51

Question 76 | Q00483

Which statement causes the loop to immediately jump to the next iteration of the loop?

A. Exit

B. Break

C. Jump

D. Continue

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer D: Continue

continue skips the rest of the current iteration and proceeds to the loop's continuation step; break exits the loop. Thus continue matches moving to the next iteration.

Why each choice is right or wrong

A - Incorrect: Exit is not the named loop-jump statement here

B - Incorrect: break leaves the loop

C - Incorrect: Jump is a category, rather than the required keyword

D - Correct: continue advances control to loop continuation

Primary note proof

Study-note section OOP25

> break exits the loop; continue skips its remaining body and then runs a for update.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 7 | lines 26-41

> Iteration: The for(), while() and
> do...while() statements
> 4. Jumping Statements: to jump out
> of loops and to control other areas of
> program java uses break and
> continue statements.
> The break statement: used to exit a
> loop
> The Continue Statement: used to
> force program to go back to the top
> of a loop
> Labels: Java does not include a goto
> statement.
> Instead of goto, Java allows you to
> combine break and continue with a
> label.

Question sources

Object-Oriented Programming practice questions · Question 53

Question 77 | Q00484

A BREAK statement inside a Loop like WHILE, FOR, DO WHILE and Enhanced-FOR causes the program execution ___ Loop

A. Exit

B. Continuation with next iteration

C. Never exit

D. Can not say

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer A: Exit

The break statement is explicitly defined as used to exit a loop.

Why each choice is right or wrong

A - Correct: Matches the stated effect

B - Incorrect: Describes continue rather than break

C - Incorrect: Contradicts the stated effect

D - Incorrect: The effect is directly stated

Primary note proof

Study-note section OOP25

> break exits the loop; continue skips its remaining body and then runs a for update.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 7 | lines 26-41

> Iteration: The for(), while() and
> do...while() statements
> 4. Jumping Statements: to jump out
> of loops and to control other areas of
> program java uses break and
> continue statements.
> The break statement: used to exit a
> loop
> The Continue Statement: used to
> force program to go back to the top
> of a loop
> Labels: Java does not include a goto
> statement.
> Instead of goto, Java allows you to
> combine break and continue with a
> label.

Question sources

Object-Oriented Programming practice questions · Question 54

Question 78 | Q00485

Java array is a collection of _____.

A. similar type of elements

B. different type of element

C. heterogeneous data

D. Both A and C

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer A: Elements with a common declared component type.

OOP26 says array elements share a declared common component type, possibly a superclass or interface. That is the precise interpretation of similar type in A. Reference elements may point to different compatible runtime subclasses, but this does not give the array unrelated declared element types or make arbitrary heterogeneous data its defining property.

Why each choice is right or wrong

A - Correct: Similar type is accurate when understood as the array's single declared common component type.

B - Incorrect: The array does not assign an unrelated declared type to each element; compatible runtime subclasses are a separate issue.

C - Incorrect: Heterogeneous runtime objects may share one component type, but unrestricted heterogeneity is not the array's defining type rule.

D - Incorrect: Combining A with unrestricted heterogeneity obscures the single declared component type required by OOP26.

Primary note proof

Study-note section OOP26

> Elements have a declared common component type, which may be a superclass/interface.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean

and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 55

Question 79 | Q00486

Array data access using _____.

A. Operator

B. Variable

C. index

D. Pointer

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer C: An index.

OOP26 defines valid array indices as zero through length-1. Selecting an array element therefore uses its index, as in a[i]. An operator participates in the syntax and a variable can hold the index, but neither term identifies the selection value as precisely. Java does not require a raw pointer to access an element.

Why each choice is right or wrong

A - Incorrect: Array-access syntax uses brackets, but operator is less specific than the index that selects the element.

B - Incorrect: A variable can store an index, but array selection also accepts other integer index expressions.

C - Correct: An index specifies the position of the requested element within the array’s valid bounds.

D - Incorrect: Java element access uses checked indices rather than requiring a programmer-visible raw pointer.

Primary note proof

Study-note section OOP26

Valid indices are 0..a.length-1; an invalid access throws ArrayIndexOutOfBoundsException.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java
(byte, short, int, long, float, double,
char & Boolean)
2. Reference types: Arrays, classes,
and interfaces, String)
• Note: Boolean type has a value true or
false (there is no conversion b/n Boolean
and other types).
• All non-primitive types are reference
types, so all class types are reference
types.
• All integers in Java are signed, Java
doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 56

Question 80 | Q00487

Java Array can allocate _____.

A. Dynamic Memory

B. Static Memory

C. Both A and B

D. None of the above

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer A: Dynamic allocation of an array object.

OOP26 shows array creation through new int[5] and describes the result as an array object with fixed length after creation. Allocation occurs when that creation expression executes, which is the dynamic-memory meaning of A. Fixed array length does not mean statically allocated storage, and a static reference does not change the array object into static memory.

Why each choice is right or wrong

A - Correct: The creation expression allocates an array object when executed, even though the created object’s length is fixed.

B - Incorrect: A fixed length does not imply static allocation; new creates an array object during execution.

C - Incorrect: Declaring a static reference does not supply a second static-allocation mechanism for the array object itself.

D - Incorrect: The valid new-array expression in OOP26 demonstrates dynamic allocation, so none is false.

Primary note proof

Study-note section OOP26

int[] a=new int[5]; and int a[]={1,2,3}; are valid. int[] a={}; creates a zero-length array; explicit new is unnecessary in a declaration initializer. int a[],b; makes only a an array; int[] c,d; makes both arrays. Array objects have fixed length after creation, although the reference may be reassigned.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java
(byte, short, int, long, float, double,
char & Boolean)
2. Reference types: Arrays, classes,
and interfaces, String)
• Note: Boolean type has a value true or
false (there is no conversion b/n Boolean
and other types).
• All non-primitive types are reference
types, so all class types are reference
types.
• All integers in Java are signed, Java
doesn‟t support unsigned integers.

Object Oriented Programming module | PDF page 8 | lines 43-55

• a class declaration only creates a
template; it does not create an actual
object.
• To actually create a Dog object, use
the second statement.
• Dog myDog; // Creates only
the template
• myDog = new Dog(); //
create a Dog object called
myDog
• After this statement executes,

myDog will be an instance of
Dog.

Question sources

Object-Oriented Programming practice questions · Question 57

Question 81 | Q00488

Which complete Java array declaration has an incorrect initializer syntax? Treat each option independently.

A. int[] arr = new int[5];

B. int arr[] = new int[5];

C. int arr[] = new int[5];

D. int arr[] = int[5] new;

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer D: int arr[] = int[5] new;

Java permits array brackets on the declared type or after the variable name, and new int[5] allocates a five-element array. Thus A is valid, and B and its duplicate C are also valid declarations. D reverses the required allocation token order, putting new after int[5]. Replacing editorial periods with semicolons isolates the intended initializer-syntax error without treating punctuation noise as the tested concept.

Why each choice is right or wrong

A - Incorrect: Brackets on the type and new int[5] are valid.

B - Incorrect: Brackets after the variable name are legal with this allocation expression.

C - Incorrect: This duplicates B's valid declaration and is not an incorrect initializer.

D - Correct: The initializer must place new before int[5], so this token order is incorrect.

Primary note proof

Study-note section OOP26

int[] a=new int[5]; and int a[]={1,2,3}; are valid. int[] a={}; creates a zero-length array; explicit new is unnecessary in a declaration initializer. int a[],b; makes only a an array; int[] c,d; makes both arrays.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 58

Question 82 | Q00489

Index in array start with _____.

A. -1

B. 0

C. 1

D. infinite

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer B: 0.

OOP26 states that valid array indices run from zero to length minus one. Thus the first element of any nonempty Java array has index 0. An empty array has no valid element index, but that does not change the language’s zero-based indexing convention requested by this question.

Why each choice is right or wrong

A - Incorrect: Negative one is below the valid lower bound and causes an invalid access when used.

B - Correct: Zero is the first valid index of a nonempty Java array.

C - Incorrect: One selects the second element when present; Java arrays do not use one-based indexing.

D - Incorrect: Infinity is not an array index; valid positions are finite integers below the fixed array length.

Primary note proof

Study-note section OOP26

> Valid indices are 0..a.length-1; an invalid access throws ArrayIndexOutOfBoundsException.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 28 | lines 65-79

> ✓ Unchecked exceptions: Is an exception that occurs at the time of execution, these are also called as Runtime Exceptions.
> ✓ These include programming bugs, such as logic errors or improper use of an API.
> ✓ Runtime exceptions are ignored at the time of compilation.
> ✓ For example, if you have declared an array of size 5 in your program, and trying to call the 6th element of the array then an ArrayIndexOutOfBoundsExceptione xception occurs.

Question sources

Object-Oriented Programming practice questions · Question 59

Question 83 | Q00490

Which shown Java declarations create and initialize a one-dimensional int array? Zero-length arrays count. Treat the options independently. Select all that apply.

A. int arr [] [] = {1, 2, 3, 4};

B. int [] arr = (1, 2, 3);

C. int [] arr = {};

D. int arr [] = {1, 2, 3};

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer C, D: int [] arr = {};; int arr [] = {1, 2, 3};

C creates an int array of length zero, and D creates one of length three initialized to 1, 2 and 3. Both declare, create and initialize a one-dimensional array. A declares two dimensions but supplies scalar elements instead of row arrays. B uses parentheses, which do not form an array initializer. Explicitly allowing zero length prevents incorrectly rejecting the valid empty array.

Why each choice is right or wrong

A - Incorrect: The flat int values do not initialize the declared int[][] component rows.

B - Incorrect: Parentheses do not provide a Java array initializer.

C - Correct: Empty braces validly create and initialize a zero-length int array.

D - Correct: The brace list validly creates and initializes a three-element int array.

Primary note proof

Study-note section OOP26

> int[] a=new int[5]; and int a[]={1,2,3}; are valid. int[] a={}; creates a zero-length array; explicit new is unnecessary in a declaration initializer. int a[],b; makes only a an array; int[] c,d; makes both arrays.

Study-note section OOP26

> Java two-dimensional arrays are arrays of arrays and may be ragged: {{12,-9,8},{7,14},{-32,-1,0}} has row lengths 3,2,3.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean
> and other types).
> • All non-primitive types are reference
> types, so all class types are reference
> types.
> • All integers in Java are signed, Java
> doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 60

Question 84 | Q00491

We can calculate the length of an array using _____.

A. sizeof(array)

B. array.len

C. array.length

D. array.sizeof()

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer C: array.length.

OOP26 uses a.length as the fixed element count when defining valid indices. Therefore array.length is the correct Java expression for the length field of an array reference named array. This is a field access rather than a method invocation, and the other supplied spellings are not the array-length mechanism described in the note.

Why each choice is right or wrong

A - Incorrect: sizeof is not the Java array-length expression used in OOP26's index-bound rule.

B - Incorrect: The field name is length, not len, so this spelling does not retrieve array length.

C - Correct: Java arrays expose their fixed element count through the length field, without parentheses.

D - Incorrect: The note uses a length field; arrays do not expose the proposed sizeof method.

Primary note proof

Study-note section OOP26

> Array objects have fixed length after creation, although the reference may be reassigned.

Study-note section OOP26

> Valid indices are 0..a.length-1; an invalid access throws ArrayIndexOutOfBoundsException.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean

and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 61

Question 85 | Q00492

Which is an inherent property of Java arrays? Here “Code Optimization” means a guarantee that merely introducing an array makes every algorithm run faster.

A. Code Optimization

B. Random access

C. Size No-Limit

D. Both A and B

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer B: Random access is the concrete array advantage.

Arrays provide indexed random access, so B states the inherent capability. They have a fixed length after creation and are bounded by implementation limits and available resources, so unlimited size is false. An array is a storage structure, not a guarantee that every algorithm becomes faster: allocating an unused array cannot improve a computation that does not need it. A therefore fails the stated guarantee, and combining it with B does not make D correct.

Why each choice is right or wrong

A - Incorrect: The stated universal speedup does not follow from array storage; adding an unused allocation does not improve the algorithm’s required computation.

B - Correct: Indexed random access is a direct array capability.

C - Incorrect: Array length is fixed after creation and constrained by implementation limits and resources.

D - Incorrect: This conjunction includes the false universal optimization guarantee, even though random access itself is true.

Primary note proof

Study-note section OOP27

Random access is the reliable array advantage; size is bounded by implementation/resources, not unlimited.

Study-note section OOP26

Array objects have fixed length after creation, although the reference may be reassigned.

Study-note section OOP03

Student s=new Student(); allocates and initializes an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java
(byte, short, int, long, float, double,
char & Boolean)
2. Reference types: Arrays, classes,
and interfaces, String)
• Note: Boolean type has a value true or
false (there is no conversion b/n Boolean
and other types).
• All non-primitive types are reference
types, so all class types are reference
types.
• All integers in Java are signed, Java
doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 62

Question 86 | Q00493

Which term describes the logical index order of a Java array, from 0 through length−1, rather than making any claim about physical RAM placement?

A. Random

B. Sequential

C. Sequential & Random

D. Binary search

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer B: Sequential

Java arrays expose a logical sequence of indices starting at zero and ending at length minus one. Sequential describes that order. Random access describes the ability to choose an index directly, not a random logical ordering of elements or a portable physical-memory arrangement. Binary search is an algorithm rather than an ordering description. The revised question therefore tests index semantics without assuming exposed physical contiguity.

Why each choice is right or wrong

A - Incorrect: Random access does not mean that the logical index sequence is randomly ordered.

B - Correct: Sequential describes the logical index progression from 0 to length−1.

C - Incorrect: Combining access capability with ordering does not give the precise ordering term requested.

D - Incorrect: Binary search is an algorithm, not the array's logical index order.

Primary note proof

Study-note section OOP26

> Valid indices are 0..a.length-1; an invalid access throws ArrayIndexOutOfBoundsException.

Study-note section OOP27

> Random access is the reliable array advantage; size is bounded by implementation/resources, not unlimited.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean
> and other types).
> • All non-primitive types are reference
> types, so all class types are reference
> types.
> • All integers in Java are signed, Java
> doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 63

Question 87 | Q00494

What will be the output of the program?

```
class Main
{
public static void main(String args[]) {
int arr[] = {10, 20, 30, 40, 50};
for(int i=0; i < arr.length; i++)
{
System.out.print(" " + arr[i]);
}
}
}
```

A. 10 20 30 40 50

B. Compiler Error

C. 10 20 30 40

D. None of the above

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer D: Exact output: " 10 20 30 40 50", with an initial space.

The loop visits all five indices, 0 through 4. Each print prepends the literal string containing one space, including before the first element. The exact output is therefore " 10 20 30 40 50" without a newline. A lists the correct numbers but omits that initial space. Under literal output matching, D is the offered exact answer.

Why each choice is right or wrong

A - Incorrect: The listed values are right, but the printed option omits the leading space emitted before 10.

B - Incorrect: The array declaration and traversal stay within indices 0 through 4, so no compilation error follows.

C - Incorrect: The last index 4 is visited and prints 50; this also omits the initial space.

D - Correct: None of the other options reproduces the exact leading-space output required by the literal code.

Primary note proof

Study-note section OOP26

> int[] a=new int[5]; and int a[]={1,2,3}; are valid.

Study-note section OOP26

> Valid indices are 0..a.length-1; an invalid access throws ArrayIndexOutOfBoundsException.

Study-note section OOP25

while and for pre-test and may execute zero times; do-while post-tests and executes once.

Study-note section OOP24

Once a String operand is encountered, + concatenates in left-to-right groups: 1+2+"x" gives "3x", while "x"+1+2 gives "x12".

Study-note section OOP-S02

PrintStream.print(double) converts the stored value through String.valueOf and Double.toString. For finite nonzero magnitudes from 0.001 inclusive to 10000000 exclusive, the standard decimal form uses at least one fractional digit and only the further digits needed to distinguish the value from adjacent doubles. println(double) also adds a line terminator. This is not fixed-decimal-place formatting.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java
(byte, short, int, long, float, double,
char & Boolean)
2. Reference types: Arrays, classes,
and interfaces, String)
• Note: Boolean type has a value true or
false (there is no conversion b/n Boolean
and other types).
• All non-primitive types are reference
types, so all class types are reference
types.
• All integers in Java are signed, Java
doesn‟t support unsigned integers.

Object Oriented Programming module | PDF page 7 | lines 26-41

Iteration: The for(), while() and
do...while() statements
4. Jumping Statements: to jump out
of loops and to control other areas of
program java uses break and
continue statements.
The break statement: used to exit a
loop
The Continue Statement: used to
force program to go back to the top
of a loop
Labels: Java does not include a goto

statement.
Instead of goto, Java allows you to
combine break and continue with a
label.

Question sources

Object-Oriented Programming practice questions · Question 64

Question 88 | Q00495

Which of these is an incorrect Statement?

A. It is necessary to use new operator to initialize an array

B. Array can be initialized using comma separated expressions surrounded by curly braces

C. Array can be initialized when they are declared

D. None of the mentioned

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer A: Explicit new is not necessary in a declaration initializer.

OOP26 explicitly states that new need not be written in a declaration initializer and shows both empty and nonempty brace forms. Thus A is the incorrect statement. B and C describe the valid brace-initializer form used at declaration. This does not imply that an existing array variable can later be assigned a bare brace list.

Why each choice is right or wrong

A - Correct: The claim is false: brace initialization at declaration creates an array without explicitly writing new.

B - Incorrect: This is a true description of the comma-separated brace initializer shown in OOP26.

C - Incorrect: This is true: the note's examples initialize arrays as part of their declarations.

D - Incorrect: A is an offered false statement, so it is incorrect to say none is incorrect.

Primary note proof

Study-note section OOP26

int[] a=new int[5]; and int a[]={1,2,3}; are valid. int[] a={}; creates a zero-length array; explicit new is unnecessary in a declaration initializer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 65

Question 89 | Q00496

What is the type of variable "b" and "d" in the below snippet?
int a[], b;
int []c, d;

A. "b" and "d" are int

B. "b" and "d" are arrays of type int

C. "d" is int variable; and "b" is int array

D. "b" is int variable; and "d" is int array

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer D: b is int; d is int[].

In int a[],b, the brackets belong only to a, leaving b as a scalar int. In int[] c,d, the brackets are part of the common type, so both c and d are int arrays. OOP26 gives these exact forms. The distinction depends on bracket placement rather than on the variable names.

Why each choice is right or wrong

A - Incorrect: b is scalar int, but d belongs to the common int[] type of the second declaration.

B - Incorrect: Only d is an array; the first declaration attaches brackets to a rather than to b.

C - Incorrect: This reverses the actual types: b is scalar int and d is int[].

D - Correct: Brackets after a leave b scalar, while brackets after int make both c and d arrays.

Primary note proof

Study-note section OOP26

int a[],b; makes only a an array; int[] c,d; makes both arrays.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java
(byte, short, int, long, float, double,
char & Boolean)
2. Reference types: Arrays, classes,
and interfaces, String)
• Note: Boolean type has a value true or
false (there is no conversion b/n Boolean
and other types).
• All non-primitive types are reference
types, so all class types are reference
types.
• All integers in Java are signed, Java
doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 66

Question 90 | Q00497

How to sort an array?

A. Array.sort()

B. Arrays.sort()

C. Collection.sort()

D. System.sort()

Hint

Trace every element or API argument and distinguish copying values from aliasing a reference.

Answer B: Arrays.sort(array).

OOP27 specifies java.util.Arrays.sort(a) for in-place sorting of an array. B names that method, with the argument omitted as shorthand in the options; a real invocation supplies the array. The other class/method combinations do not match the array-sorting API provided here.

Why each choice is right or wrong

A - Incorrect: The supplied class name Array does not match the java.util.Arrays class specified here.

B - Correct: Arrays.sort is the correct method name; actual use supplies the array argument as Arrays.sort(a).

C - Incorrect: Collection.sort is not the array-sorting API identified in OOP27.

D - Incorrect: System.sort is not the array-sorting API provided in the complete note.

Primary note proof

Study-note section OOP27

> java.util.Arrays.sort(a) sorts in place.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean
> and other types).
> • All non-primitive types are reference
> types, so all class types are reference
> types.
> • All integers in Java are signed, Java
> doesn‟t support unsigned integers.

Object Oriented Programming module | PDF page 2 | lines 42-45

> • The API provides libraries, which
> groups a number of related classes
> and interfaces together into what is
> called a Java package.

Question sources

Object-Oriented Programming practice questions · Question 67

Question 91 | Q00498

Which correctly spelled Java API copies a selected range from one existing array into another existing array? The parentheses in distractors stand for a method name, not a complete no-argument invocation.

A. System.arraycopy(src, srcPos, dst, dstPos, length)

B. Array.copy()

C. Arrays.copy()

D. Collection.copy()

Hint

Trace every element or API argument and distinguish copying values from aliasing a reference.

Answer A: System.arraycopy(src, srcPos, dst, dstPos, length)

System.arraycopy copies length elements from the supplied source position into a supplied destination array position. Its name uses lowercase c in copy. Arrays.copyOf is a different supported API that creates a copy, but Arrays.copy is not that method's name. The other printed names do not identify the requested array-copy API. Correcting A's spelling and supplying its arguments restores a usable exact choice.

Why each choice is right or wrong

A - Correct: This is the correctly spelled range-copy API for existing source and destination arrays.

B - Incorrect: Array.copy is not the supported API named here.

C - Incorrect: The supported creation API is Arrays.copyOf, not Arrays.copy.

D - Incorrect: Collection.copy is not the specified array-range-copy API.

Primary note proof

Study-note section OOP27

> System.arraycopy(src,srcPos,dst,dstPos,length) copies into an existing array; its exact spelling uses lowercase c. Arrays.copyOf(a,n) creates a copy. a=b merely aliases the same array.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean

and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Object-Oriented Programming practice questions · Question 68

Question 92 | Q00499

Which named type categories can be directly declared as top-level members of a Java package? Subpackages are separately named packages, not automatically enclosed members. Select all that apply.

A. Classes

B. Sub Packages

C. Interfaces

D. All of the above

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer A, C: Classes; Interfaces

A package organizes named types, including classes and interfaces, and provides a naming and access boundary. A subpackage is a separate package identity even when its name starts with the parent’s name. It does not automatically gain membership or access through that spelling. Consequently classes and interfaces are the direct type categories, while the all option wrongly includes subpackages in the same membership sense.

Why each choice is right or wrong

A - Correct: Classes can be top-level named types belonging to the package.

B - Incorrect: A subpackage is a separate named package rather than a directly declared type member.

C - Correct: Interfaces can be top-level named types belonging to the package.

D - Incorrect: All includes the separate-package category and is therefore too broad.

Primary note proof

Study-note section OOP28

package p; precedes imports; imports precede top-level type declarations. import p.*; imports accessible types from p as needed, not types in p.sub automatically. Packages organize names

and control access; hierarchical spelling does not grant special access between a package and its subpackages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 42-45

• The API provides libraries, which groups a number of related classes and interfaces together into what is called a Java package.

Question sources

Object-Oriented Programming practice questions · Question 69

Question 93 | Q00500

Which of these keywords is used to define packages in Java?

A. pkg

B. Pkg

C. package

D. Package

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer C: package

The source-file rules refer to a package statement as the first statement when a class belongs to a package; the provided spelling is package.

Why each choice is right or wrong

A - Incorrect: pkg is not the package-declaration keyword

B - Incorrect: Pkg is not the stated spelling

C - Correct: Matches the package statement spelling

D - Incorrect: Capitalized Package differs from the stated spelling

Primary note proof

Study-note section OOP28

package p; precedes imports; imports precede top-level type declarations.

Study-note section OOP28

Java identifiers are case sensitive; $ is allowed but @ is not an identifier character.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 51-70

2.3. Source file declaration
rules
• These rules are essential when declaring classes, import statements and package statements in a source file.
1. There can be only one public class per source file.
2. A source file can have multiple nonpublic classes.
3. The public class name should be the name of the source file as well which should be appended by .java at the end.
For example: the class name is public class Employee{} then the source file should be as Employee.java.
• If the class is defined inside a package, then the package statement should be the first statement in the source file.

Question sources

Object-Oriented Programming practice questions · Question 70

Question 94 | Q00501

An _____ statement can be used to access the classes and interface of a different package from the current package.

A. instanceOf

B. import

C. extends

D. implement

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer B: import

Import statements follow any package declaration and precede top-level type declarations; they come first when there is no package declaration. They bring external class/interface names into the source context.

Why each choice is right or wrong

A - Incorrect: instanceof is an object-type test, not an import

B - Correct: Matches the source-file import statement

C - Incorrect: extends defines inheritance

D - Incorrect: implement is not the import statement

Primary note proof

Study-note section OOP28

> package p; precedes imports; imports precede top-level type declarations. import p.*; imports accessible types from p as needed, not types in p.sub automatically. Packages organize names and control access; hierarchical spelling does not grant special access between a package and its subpackages.

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Study-note section OOP21

> Use if(a instanceof Ferrari) before narrowing when the dynamic type is uncertain.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 51-70

> 2.3. Source file declaration
> rules
> • These rules are essential when declaring
> classes, import statements and package
> statements in a source file.
> 1. There can be only one public
> class per source file.
> 2. A source file can have multiple
> nonpublic classes.
> 3. The public class name should be
> the name of the source file as
> well which should be appended
> by .java at the end.
> For example: the class name is public class
> Employee{} then the source file should be
> as Employee.java.
> • If the class is defined inside a
> package, then the package statement
> should be the first statement in the
> source file.

Object Oriented Programming module | PDF page 10 | lines 2-12

package statement and the class declaration.
• If there are no package statements then the import statement should be the first line in the source file.
• Import and package statements will imply to all the classes present in the source file.
• It is not possible to declare different import and/or package statements to different classes in the source file.

Question sources

Object-Oriented Programming practice questions · Question 71

Question 95 | Q00502

Which of the following packages is used to includes classes to create user interface like Button and Checkbox?

A. java.lang

B. java.net

C. java.awt

D. java.io

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer C: java.awt.

OOP28 explicitly identifies java.awt as including Button and Checkbox. The question names those UI classes, so the answer follows directly from their package membership. java.lang, networking and I/O packages serve other purposes and are not the package the note identifies for these two AWT controls.

Why each choice is right or wrong

A - Incorrect: java.lang is implicitly imported, but OOP28 locates Button and Checkbox in java.awt instead.

B - Incorrect: The named controls are AWT UI types, not networking types in the proposed java.net package.

C - Correct: OOP28 explicitly lists Button and Checkbox as classes included in java.awt.

D - Incorrect: java.io supplies input/output facilities rather than the two AWT controls named by the question.

Primary note proof

Study-note section OOP28

java.lang is implicitly imported; java.util supplies utilities; java.io supplies I/O; java.awt includes Button and Checkbox.

Study-note section OOP-S04

Calendar and Date are classes in java.util; Collection is an interface in that package. java.util also contains Arrays and Scanner. java.net supplies networking facilities; java.awt supplies the Button and Checkbox UI controls described in the package overview.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 27 | lines 20-41

• An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.
• An exception is an indication of a problem that occurs during a program‟s execution.
• An exception is a runtime error.
• A Java exception is an instance of a class derived from Throwable.
• The Throwable class is contained in the java.lang package, and subclasses of Throwable are contained in various packages.
• Errors related to GUI components are included in the java.awt package; numeric exceptions are included in the java.lang package because they are related to the java.lang.Number class.
• You can create your own exception classes by extending Throwable or a subclass of Throwable.

Question sources

Object-Oriented Programming practice questions · Question 72

Question 96 | Q00503

Which of the following packages is used to includes utility classes like Calendar, Collection, Date?

A. java.lang

B. java.net

C. java.awt

D. java.util

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer D: java.util.

The question identifies Calendar, Collection and Date as utility types. OOP28 assigns utility facilities to java.util, in contrast to implicitly imported language facilities, AWT user-interface classes and I/O. Therefore java.util is the package category supported for the named utility examples, rather than the language, networking or UI alternatives.

Why each choice is right or wrong

A - Incorrect: java.lang supplies core language facilities; the utility package identified here is java.util.

B - Incorrect: java.net is the networking alternative, not the utility package identified in OOP28.

C - Incorrect: java.awt contains AWT UI controls, rather than the utility facilities requested by the question.

D - Correct: java.util is the note's explicit utility-package mapping for the class/interface examples in the stem.

Primary note proof

Study-note section OOP-S04

> Calendar and Date are classes in java.util; Collection is an interface in that package. java.util also contains Arrays and Scanner. java.net supplies networking facilities; java.awt supplies the Button and Checkbox UI controls described in the package overview.

Study-note section OOP28

> java.lang is implicitly imported; java.util supplies utilities; java.io supplies I/O; java.awt includes Button and Checkbox.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 42-45

> • The API provides libraries, which
> groups a number of related classes

and interfaces together into what is
called a Java package.

Question sources

Object-Oriented Programming practice questions · Question 73

Question 97 | Q00504

An accessible class declares an ordinary field. Which offered modifier choices permit another unrelated class in the same package to access that field directly? Choose the most complete answer; this asks about member modifiers, not top-level class modifiers.

A. Public

B. Protected

C. No Modifier

D. All of the mentioned

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer D: All of the mentioned

For ordinary members, public allows access, protected includes same-package access, and no modifier grants package access. All three listed choices therefore permit the unrelated same-package client to access the field. This does not claim that protected is legal on a top-level class, and class accessibility alone would not expose a private field. Under the explicitly member-level scope, D is the complete selection.

Why each choice is right or wrong

A - Incorrect: Public permits the access, but this is only one of the three valid modifier choices.

B - Incorrect: Protected includes same-package access, but this option alone is incomplete.

C - Incorrect: No modifier grants package access, but this option alone is incomplete.

D - Correct: All three listed member-modifier choices permit the specified same-package access.

Primary note proof

Study-note section OOP08

For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP08

protected therefore does not mean “subclasses only”: unrelated classes in the same package also have access.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

Access Specifiers
Access level modifiers determine whether other classes can use a particular field or invoke a particular method.
Their are four types access levels:
1. No Modifier(Default): Visible to the package(source file), and it is called package-private
2. Private: Visible to the class only.
3. Public: Visible to the world.
4. Protected: Visible to the package and all subclasses.

Object Oriented Programming module | PDF page 9 | lines 51-70

2.3. Source file declaration rules
• These rules are essential when declaring classes, import statements and package statements in a source file.
1. There can be only one public class per source file.
2. A source file can have multiple nonpublic classes.
3. The public class name should be the name of the source file as well which should be appended by .java at the end.
For example: the class name is public class Employee{} then the source file should be as Employee.java.
• If the class is defined inside a package, then the package statement should be the first statement in the source file.

Question sources

Object-Oriented Programming practice questions · Question 74

Question 98 | Q00505

Which option is a syntactically complete Java on-demand import of accessible top-level types from package pkg, without importing subpackages automatically?

A. import pkg.

B. Import pkg.

C. import pkg.*;

D. Import pkg.*

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer C: import pkg.*;

A type-import-on-demand statement uses lowercase import, the package name, a dot and wildcard, and a semicolon. import pkg.*; therefore supplies the complete syntax. It makes accessible types from pkg available as needed without automatically importing types from pkg.sub. A lacks the wildcard or type name, and capitalized Import is not the Java keyword in B or D.

Why each choice is right or wrong

A - Incorrect: The trailing dot lacks a wildcard or type name and a complete terminator.

B - Incorrect: Capitalized Import is not the keyword, and the target is incomplete.

C - Correct: This is the complete lowercase wildcard import statement.

D - Incorrect: Capitalized Import is not a Java keyword and the statement lacks its semicolon.

Primary note proof

Study-note section OOP28

> import p.*; imports accessible types from p as needed, not types in p.sub automatically.

Study-note section OOP28

> Java identifiers are case sensitive; $ is allowed but @ is not an identifier character.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 10 | lines 2-12

> package statement and the class
> declaration.
> • If there are no package statements
> then the import statement should be
> the first line in the source file.
> • Import and package statements will
> imply to all the classes present in the

source file.
• It is not possible to declare different import and/or package statements to different classes in the source file.

Object Oriented Programming module | PDF page 27 | lines 20-41

• An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.
• An exception is an indication of a problem that occurs during a program‟s execution.
• An exception is a runtime error.
• A Java exception is an instance of a class derived from Throwable.
• The Throwable class is contained in the java.lang package, and subclasses of Throwable are contained in various packages.
• Errors related to GUI components are included in the java.awt package; numeric exceptions are included in the java.lang package because they are related to the java.lang.Number class.
• You can create your own exception classes by extending Throwable or a subclass of Throwable.

Question sources

Object-Oriented Programming practice questions · Question 75

Question 99 | Q00506

Which of the following is false statement about package in java?

A. Packages are used for preventing naming conflicts

B. Providing controlled access: protected and default have package level access control.

C. Packages cannot be considered as data encapsulation

D. Both B and C

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer C: The claim that packages cannot support encapsulation is false.

OOP28 says packages organize names and control access, while OOP08 defines package access and same-package protected access. These mechanisms can hide implementation types or members behind permitted boundaries, so C's categorical denial is false. A and B describe real package roles. D incorrectly includes the true package-access statement B among the false claims.

Why each choice is right or wrong

A - Incorrect: Organizing names into packages supports avoiding naming conflicts, so this is a true purpose rather than the false claim.

B - Incorrect: Default access is package access, and protected also grants same-package access; the statement is true.

C - Correct: Packages control access and can hide implementation types or members, contradicting this categorical denial of encapsulation.

D - Incorrect: B is supported by the access rules, so both B and C cannot be classified as false.

Primary note proof

Study-note section OOP28

> Packages organize names and control access; hierarchical spelling does not grant special access between a package and its subpackages.

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP08

> protected therefore does not mean "subclasses only": unrelated classes in the same package also have access.

Study-note section OOP02

> Information hiding keeps implementation decisions behind a stable interface.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 42-45

> • The API provides libraries, which
> groups a number of related classes
> and interfaces together into what is
> called a Java package.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether

other classes can use a particular field or
invoke a particular method.
Their are four types access levels:
1. No Modifier(Default): Visible to
the package(source file), and it is
called package-private
2. Private: Visible to the class only.
3. Public: Visible to the world.
4. Protected: Visible to the package
and all subclasses.

Question sources

Object-Oriented Programming practice questions · Question 76

Question 100 | Q00507

For a Java naming hierarchy such as p and p.q, what is the specific conventional term for packages such as p.q relative to p?

A. packages

B. nested packages sharing their parent's member-access scope

C. util subpackages

D. subpackages

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer D: Subpackages.

The conventional term for p.q relative to p is subpackage. This is a naming hierarchy, not a lexical enclosing scope: the parent's import does not automatically import types from the subpackage, and their related names grant no special member access. Packages is only the broad category. The util qualifier would restrict the answer to a particular package family instead of naming the general relationship.

Why each choice is right or wrong

A - Incorrect: Packages names the broad category but does not identify the requested relationship to a parent name.

B - Incorrect: The naming hierarchy does not give a subpackage its parent's member-access scope.

C - Incorrect: The relationship applies to arbitrary package names, not just util subpackages.

D - Correct: Subpackages is the conventional term for the requested naming relationship.

Primary note proof

Study-note section OOP28

import p.*; imports accessible types from p as needed, not types in p.sub automatically. Packages organize names and control access; hierarchical spelling does not grant special access between a package and its subpackages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 42-45

• The API provides libraries, which groups a number of related classes and interfaces together into what is called a Java package.

Question sources

Object-Oriented Programming practice questions · Question 77

Question 101 | Q00508

_____ can be defined as the process where one class acquires the properties (methods and fields) of another.

A. Overriding

B. Inheritance

C. Polymorphism

D. Abstraction

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer B: Inheritance

Inheritance means acquiring another class's methods and fields.

Why each choice is right or wrong

A - Incorrect: Overrides a method rather than acquiring the class properties generally

B - Correct: Matches the definition word for word

C - Incorrect: Many forms is a different principle

D - Incorrect: Exposes essential information rather than acquiring class properties

Primary note proof

Study-note section OOP13

class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Study-note section OOP13

Use inheritance when a subtype can satisfy the parent's contract: Dog is an Animal.

Study-note section OOP02

Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Study-note section OOP14

A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype.

Study-note section OOP02

Abstraction presents the essential operations while omitting details irrelevant to the caller.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{

```
.....}
class Sub extends Super{ }
```

Question sources

Object-Oriented Programming practice questions · Question 78

Question 102 | Q00509

The class which inherits the properties of other is known as _____

A. superclass

B. parent class

C. subclass

D. None of the above

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer C: subclass

The class that does the inheriting is explicitly called subclass, derived class or child class.

Why each choice is right or wrong

A - Incorrect: Superclass is the source class

B - Incorrect: Parent class is the source class

C - Correct: Subclass is the inheriting class

D - Incorrect: C identifies the inheriting class correctly

Primary note proof

Study-note section OOP13

class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one

class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Question sources

Object-Oriented Programming practice questions · Question 79

Question 103 | Q00510

Subclass also known as ?

A. derived class

B. child class

C. base class

D. Both A and B

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer D: Both A and B

Subclass, derived class and child class are synonyms; base class is a superclass synonym.

Why each choice is right or wrong

A - Incorrect: A valid subclass synonym

B - Incorrect: A valid subclass synonym

C - Incorrect: Names the superclass instead

D - Correct: Combines both valid synonyms

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

> What is Inheritance?
> • Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
> • Inheritance is the process where one class acquires the properties (methods and fields) of another.
> • A class that is inherited is called a superclass.
> • The class that does the inheriting is called a subclass.
> • OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
> • The class whose properties are inherited is known as superclass (base class, parent class).
> • Therefore, a subclass is a specialized version of a superclass.
> • It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
> • The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
> • Sytax:
> class Super{
> .....}
> class Sub extends Super{ }

Question sources

Object-Oriented Programming practice questions · Question 80

Question 104 | Q00511

_____ is the keyword used to inherit the properties of a class.

A. inherit

B. poly

C. extends

D. super

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer C: extends

The inheritance syntax is class Sub extends Super.

Why each choice is right or wrong

A - Incorrect: Not the stated keyword

B - Incorrect: Not the stated keyword

C - Correct: Explicitly identified inheritance keyword

D - Incorrect: Refers to the superclass rather than declaring inheritance

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Study-note section OOP16

> super.name selects a hidden superclass field; super.sound() directly requests the parent implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

> What is Inheritance?
> • Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
> • Inheritance is the process where one

class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Question sources

Object-Oriented Programming practice questions · Question 81

Question 105 | Q00512

The super keyword is similar to _____ keyword.

A. construct

B. this

C. class

D. extends

Hint

Distinguish the current instance, parent context, hidden fields and static dispatch.

Answer B: this

super accesses superclass members, while this refers to the current object; the comparison concerns member access, rather than distinct receiver objects.

Why each choice is right or wrong

A - Incorrect: construct is not the compared keyword

B - Correct: Explicitly named comparison

C - Incorrect: class defines a type rather than this-like access

D - Incorrect: extends establishes inheritance rather than member reference

Primary note proof

Study-note section OOP16

Java this denotes the current object; this.name=name distinguishes a field from a parameter. super.name selects a hidden superclass field; super.sound() directly requests the parent implementation. It does not create a second parent object. An unqualified instance call may dispatch dynamically; an explicit super call is different.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 17 | lines 51-74

The super keyword
• Since encapsulation is a primary attribute of OOP, Whenever a subclass needs to refer to its immediate superclass, it can do so by use of the keyword super.
• Super has two general forms.
1. Used to access parent class instance members.
2. Used to access parent class constructor.
1. Used to access parent class instance members.
• super acts somewhat like this, except that it always refers to the superclass of the subclass in which it is used.
• Super has the form: super.member Here, member can be either a method or an instance variable.
• Super is most applicable to resolve member hiding.
• super( ) must always be the first statement executed inside a subclass‟ constructor.

Question sources

Object-Oriented Programming practice questions · Question 82

Question 106 | Q00513

Which of these is correct way of inheriting class A by class B?

A. class B + class A {}

B. class B inherits class A {}

C. class B extends A {}

D. class B extends class A {}

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer C: class B extends A {}

The declared subclass comes first, followed by extends and the superclass name; the example is class Sub extends Super.

Why each choice is right or wrong

A - Incorrect: Plus is not the inheritance syntax

B - Incorrect: inherits is not the inheritance keyword

C - Correct: Correct order and single class keyword

D - Incorrect: Extra class before the parent violates the demonstrated form

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

> What is Inheritance?
> • Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
> • Inheritance is the process where one class acquires the properties (methods and fields) of another.
> • A class that is inherited is called a superclass.
> • The class that does the inheriting is called a subclass.

• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Question sources

Object-Oriented Programming practice questions · Question 83

Question 107 | Q00514

A class which is declared with the _____ keyword is known as an abstract class in Java.

A. abstract

B. util

C. extends

D. None of the above

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer A: abstract

The abstract-class definition explicitly uses the abstract keyword.

Why each choice is right or wrong

A - Correct: Matches the definition

B - Incorrect: util is not the modifier

C - Incorrect: extends names an inheritance relation

D - Incorrect: A directly supplies the keyword

Primary note proof

Study-note section OOP18

> An abstract class cannot be directly instantiated but can define fields, constructors, abstract methods and concrete methods. It need not contain an abstract method. An abstract method has no body; abstract and final conflict because an operation cannot require overriding while forbidding it.

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 23 | lines 57-81

> Abstract Classes
> • A classes declared with abstract
> keyword and can‟t be instantiated.
> • Used to hide meaningless methods:
> • There can be no objects of an abstract
> class. .:.
> • An abstract class cannot be directly
> instantiated with the new operator.
> • You cannot declare abstract
> constructors, or abstract static methods
> • Syntax: abstract class class_name
> Example:
> abstract class A {
> abstract void callme();
> // concrete methods are still
> allowed in abstract classes
> void callmetoo() {
> System.out.println("This is a
> concrete method.");
> }}
> class B extends A {
> void callme() {
> System.out.println("B's
> implementation of callme.");
> }}

Question sources

Object-Oriented Programming practice questions · Question 84

Question 108 | Q00515

A method which is declared as abstract and does not have implementation is known as an _____?

A. Abstract Interface

B. Abstract Thread

C. Abstract List

D. abstract Method

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer D: abstract Method

The summary says abstract methods have no body/implementation. That is the method kind described by the stem.

Why each choice is right or wrong

A - Incorrect: Confuses a method with an interface

B - Incorrect: Thread is not the unimplemented method category

C - Incorrect: List is not the unimplemented method category

D - Correct: Matches a method without implementation

Primary note proof

Study-note section OOP18

> An abstract method has no body; abstract and final conflict because an operation cannot require overriding while forbidding it.

Study-note section OOP19

> interface Payable{double getPaymentAmount();} declares an abstract public operation. class Invoice implements Payable must provide public double getPaymentAmount(), perhaps returning getQuantity()*getPricePerItem().

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 24 | lines 63-73

> Summery:
> Abstract:
> • Classes: declared with abstract
> keyword and can‟t be instantiated.
> • Methods: without body (no
> implementation)

Final:
• Class: declared as final prevents inheritance.
• Method: declared as final prevents overriding that method.

Question sources

Object-Oriented Programming practice questions · Question 85

Question 109 | Q00516

Which offered access modifier may be explicitly applied to an ordinary top-level Java interface? This is not a member interface nested in another type.

A. Public

B. private

C. Protected

D. All of the mentioned

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer A: Public

An ordinary top-level interface may be public or package-private. Of the explicit access modifiers listed, only public is permitted at top level. Private and protected can apply in some nested member contexts but are excluded by the stated declaration location. No modifier is the package-private alternative, though it is not one of the offered choices. Therefore the aggregate of all modifiers is false.

Why each choice is right or wrong

A - Correct: Public is permitted on an ordinary top-level interface.

B - Incorrect: Private is not a legal top-level interface modifier.

C - Incorrect: Protected is not a legal top-level interface modifier.

D - Incorrect: All includes the forbidden private and protected top-level modifiers.

Primary note proof

Study-note section OOP08

Top-level classes/interfaces may be public or package-private, not protected/private. Member types have additional visibility possibilities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 33 | lines 31-46

The d/c b/n Interface & Class
• You cannot instantiate an interface
• Interface has not constructor
• Methods in an interface are all abstract.
• Fields in an interface are all static and final
• An interface can extend multiple interfaces
Summery
• By default, methods in interface are public and abstract.
• Data members are public, static and final.
• An interface with empty body is called Tag/marker interface.

Question sources

Object-Oriented Programming practice questions · Question 86

Question 110 | Q00517

Which of these keywords is used by a class to use an interface defined previously?

A. Import

B. import

C. implements

D. Implements

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer C: implements

A class implements an interface, while an interface extends another interface.

Why each choice is right or wrong

A - Incorrect: Import is neither the demonstrated spelling nor implementation keyword

B - Incorrect: import is a source naming facility, not implementation

C - Correct: Matches class implementation syntax

D - Incorrect: Capitalized form is not the stated keyword

Primary note proof

Study-note section OOP19

interface Payable{double getPaymentAmount();} declares an abstract public operation. class Invoice implements Payable must provide public double getPaymentAmount(), perhaps returning getQuantity()*getPricePerItem(). A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces.

Study-note section OOP28

package p; precedes imports; imports precede top-level type declarations. import p.*; imports accessible types from p as needed, not types in p.sub automatically. Packages organize names and control access; hierarchical spelling does not grant special access between a package and its subpackages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 31 | lines 9-30

• It provides total abstraction;
• Means all the methods in interface are declared with empty body and are public and all fields are public, static and final by default.
• A class that implement interface must implement all the methods declared in the interface.
• If a class implements an interface and does not provide method bodies for all functions specified in the interface, then class must be declared abstract.
Rules of Interface
• Variables declared in an interface are public, static & final by default.
• Java does not allow you to extend more than one class, However you can implement more than one interfaces in your class.
• class implements interface but an interface extends another interfaces.

Question sources

Object-Oriented Programming practice questions · Question 87

Question 111 | Q00518

Which modifier lists can legally prefix int X = 1; when it is declared as a field in a Java interface? Missing public/static/final modifiers are supplied implicitly. Treat each choice independently. Select all that apply.

A. public static

B. private final

C. public final

D. static final

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer A, C, D: public static; public final; static final

Every interface field is implicitly public static final. A can omit final, C can omit static, and D can omit public because the language supplies each missing modifier. The initializer int X=1 satisfies the constant field's initialization requirement. Private conflicts with the field's public visibility and is therefore forbidden. The three legal partial modifier lists all produce the same effective field modifiers.

Why each choice is right or wrong

A - Correct: public static is legal because final is implicit.

B - Incorrect: private conflicts with an interface field's implicit public visibility.

C - Correct: public final is legal because static is implicit.

D - Correct: static final is legal because public is implicit.

Primary note proof

Study-note section OOP19

> No interface instance fields or constructors exist; fields are implicitly public static final.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 31 | lines 9-30

> • It provides total abstraction;
> • Means all the methods in interface
> are declared with empty body and
> are public and all fields are public,
> static and final by default.
> • A class that implement interface
> must implement all the methods
> declared in the interface.
> • If a class implements an interface

and does not provide method bodies for all functions specified in the interface, then class must be declared abstract.
Rules of Interface
• Variables declared in an interface are public, static & final by default.
• Java does not allow you to extend more than one class, However you can implement more than one interfaces in your class.
• class implements interface but an interface extends another interfaces.

Question sources

Object-Oriented Programming practice questions · Question 88

Question 112 | Q00519

In Java 17, which option most completely describes the permitted method contents of an interface? Here “declaration” means an abstract method declaration without a body and “definition” means a permitted method with a body; it does not ask what every interface must contain.

A. Method definition

B. Method declaration

C. Method declaration and definition

D. Method name

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer C: Method declaration and definition

Modern Java interfaces can contain abstract method declarations and method bodies of permitted kinds, including default and static methods and private helpers. They need not contain every kind, but both declarations and definitions are allowed. A and B each describe only part of that capability. C is the most complete category, while a bare method name does not constitute the required declaration or definition.

Why each choice is right or wrong

A - Incorrect: Permitted method bodies are part of the answer, but this omits abstract declarations.

B - Incorrect: Abstract declarations are allowed, but this omits permitted method bodies.

C - Correct: Both abstract declarations and permitted method definitions can appear in Java 17 interfaces.

D - Incorrect: A name alone is not a complete method declaration or definition.

Primary note proof

Study-note section OOP19

> A method declaration without private, default or static is implicitly public abstract. Modern interfaces may also have default/static method bodies, and Java 9+ private helper methods.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 30 | lines 37-59

> Chapter - Five: Java Interface
> • An interface in java is a blueprint of a class.
> • It has static constants and abstract methods.
> • The interface in java is a mechanism to achieve abstraction.
> • There can be only abstract methods in the java interface not method body (contain only signature, not body).
> • It is used to achieve abstraction and multiple inheritance in Java.
> • Java Interface also represents IS-A relationship.
> • It cannot be instantiated just like abstract class.
> • There are mainly three reasons to use interface.
> • It is used to achieve abstraction.
> • By interface, we can support the functionality of multiple inheritance.

Object Oriented Programming module | PDF page 33 | lines 31-46

> The d/c b/n Interface & Class
> • You cannot instantiate an interface
> • Interface has not constructor
> • Methods in an interface are all abstract.
> • Fields in an interface are all static and final
> • An interface can extend multiple

interfaces
Summery
• By default, methods in interface are public and abstract.
• Data members are public, static and final.
• An interface with empty body is called Tag/marker interface.

Question sources

Object-Oriented Programming practice questions · Question 89

Question 113 | Q00520

What type of methods an interface contain by default?

A. abstract

B. static

C. final

D. private

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer A: abstract

The summary explicitly says interface methods are public and abstract by default.

Why each choice is right or wrong

A - Correct: Matches the stated default

B - Incorrect: Not the default method category in this context

C - Incorrect: Not the stated default

D - Incorrect: Contradicts the public default

Primary note proof

Study-note section OOP19

A method declaration without private, default or static is implicitly public abstract.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 33 | lines 31-46

The d/c b/n Interface & Class
• You cannot instantiate an interface
• Interface has not constructor
• Methods in an interface are all abstract.
• Fields in an interface are all static and final
• An interface can extend multiple interfaces
Summery
• By default, methods in interface are public and abstract.
• Data members are public, static and final.
• An interface with empty body is called Tag/marker interface.

Question sources

Object-Oriented Programming practice questions · Question 90

Question 114 | Q00521

When does Exceptions in Java arises in code sequence?

A. Run Time

B. Can Occur Any Time

C. Compilation Time

D. None of the mentioned

Hint

Separate source errors from runtime exceptions and determine the hierarchy and checked obligation.

Answer A: Exceptions are thrown at runtime; checked obligations are checked at compile time.

OOP29 explains that checked does not mean an exception occurs during compilation. The compiler verifies catch-or-declare obligations; an actual exception throw occurs during program execution. Therefore Run Time is the correct occurrence answer. Compile-time syntax or name errors are a different category and do not make Java exceptions occur at any arbitrary stage.

Why each choice is right or wrong

A - Correct: An actual exception throw occurs during execution, as OOP29 explicitly states.

B - Incorrect: Any time conflates runtime throwing with compile-time checking and source errors.

C - Incorrect: Compilation checks obligations or reports source errors; it is not when the described program throws an exception.

D - Incorrect: Run Time is offered and is the supported occurrence stage, so none is false.

Primary note proof

Study-note section OOP29

A missing semicolon or undeclared variable is a compile-time source error, not a checked exception.

Study-note section OOP29

Checked does not mean the exception occurs at compile time: the compiler checks an obligation to catch or declare it, while an actual throw occurs at runtime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 27 | lines 20-41

• An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.
• An exception is an indication of a problem that occurs during a program‟s execution.
• An exception is a runtime error.
• A Java exception is an instance of a class derived from Throwable.
• The Throwable class is contained in the java.lang package, and subclasses of Throwable are contained in various packages.
• Errors related to GUI components are included in the java.awt package; numeric exceptions are included in the java.lang package because they are related to the java.lang.Number class.
• You can create your own exception classes by extending Throwable or a subclass of Throwable.

Object Oriented Programming module | PDF page 28 | lines 25-43

1. Checked exception: Is an exception that occurs at the compile time,
✓ These are also called as compile time exceptions.

✓ These exceptions cannot simply be ignored at the time of compilation,
✓ The Programmer should take care of (handle) these exceptions.
✓ For example, if you use FileReader class in your program to read data from a file, if the file specified in its constructor doesn't exist, then a FileNotFoundException occurs, and compiler prompts the programmer to handle the exception.

Object Oriented Programming module | PDF page 28 | lines 65-79

✓ Unchecked exceptions: Is an exception that occurs at the time of execution, these are also called as Runtime Exceptions.
✓ These include programming bugs, such as logic errors or improper use of an API.
✓ Runtime exceptions are ignored at the time of compilation.
✓ For example, if you have declared an array of size 5 in your program, and trying to call the 6th element of the array then an ArrayIndexOutOfBoundsExceptione xception occurs.

Question sources

Object-Oriented Programming practice questions · Question 91

Question 115 | Q00522

Which of these keywords is not a part of exception handling?

A. finally

B. thrown

C. catch

D. try

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer B: thrown

try, catch and finally are Java exception-handling keywords; thrown is an English word in compiler diagnostics, rather than a Java keyword.

Why each choice is right or wrong

A - Incorrect: Present in the example's cleanup block

B - Correct: Not a keyword used to declare or handle exceptions; distinguish the word thrown in diagnostics

C - Incorrect: Present as the handler clause

D - Incorrect: Present as the monitored block

Primary note proof

Study-note section OOP30

> throw new IOException("message") actively throws a value; throws IOException in a signature declares a possible checked exception. A caller must handle or propagate checked obligations. try encloses guarded operations; a matching catch handles the thrown type or a subtype. Execution skips the remainder of the try when a throw occurs.

Study-note section OOP31

> finally normally executes after try and any selected catch, including when leaving by return or propagating an exception.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 28 | lines 8-22

```
try {
int num=
reader.nextInt();
System.out.println("The number
is :" + num);
}
catch
(InputMismatchException e){
System.out.println("There is
type mis match :");
}
finally{
System.out.println("This part is
always done");
}} }
```

Object Oriented Programming module | PDF page 29 | lines 11-22

```
Using try and catch…
General form of try/catch:
Try{
//block of code to monitor for
errors
}
Catch(ExceptionType1 exOb ) {
//handler for ExceptionType1
}
Catch(ExceptionType2 exOb ) {
//handler for ExceptionType2
}
```

Object Oriented Programming module | PDF page 28 | lines 25-43

1. Checked exception: Is an exception that occurs at the compile time,
✓ These are also called as compile time exceptions.
✓ These exceptions cannot simply be ignored at the time of compilation,
✓ The Programmer should take care of (handle) these exceptions.
✓ For example, if you use FileReader class in your program to read data from a file, if the file specified in its constructor doesn't exist, then a FileNotFoundException occurs, and compiler prompts the programmer to handle the exception.

Question sources

Object-Oriented Programming practice questions · Question 92

Object-Oriented Programming practice questions · Question 93

Question 116 | Q00524

Which of these keywords must be used to handle the exception thrown by try block in some rational manner?

A. finally

B. throw

C. catch

D. try

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer C: catch

The catch clause contains the handler for its exception type. finally is separate cleanup, and try surrounds the monitored operation.

Why each choice is right or wrong

A - Incorrect: Runs final code rather than the type-specific handler

B - Incorrect: Would signal an exception rather than handle one

C - Correct: The explicit exception handler clause

D - Incorrect: Marks monitored code, not the handler

Primary note proof

Study-note section OOP30

> throw new IOException("message") actively throws a value; throws IOException in a signature declares a possible checked exception. A caller must handle or propagate checked obligations. try encloses guarded operations; a matching catch handles the thrown type or a subtype. Execution skips the remainder of the try when a throw occurs.

Study-note section OOP31

> finally normally executes after try and any selected catch, including when leaving by return or propagating an exception.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 28 | lines 8-22

```
try {
int num=
reader.nextInt();
System.out.println("The number
is :" + num);
}
```

```
catch
(InputMismatchException e){
System.out.println("There is
type mis match :");
}
finally{
System.out.println("This part is
always done");
}} }
```

Object Oriented Programming module | PDF page 29 | lines 11-22

```
Using try and catch...
General form of try/catch:
Try{
//block of code to monitor for
errors
}
Catch(ExceptionType1 exOb ) {
//handler for ExceptionType1
}
Catch(ExceptionType2 exOb ) {
//handler for ExceptionType2
}
```

Question sources

Object-Oriented Programming practice questions · Question 94

Object-Oriented Programming practice questions · Question 95

Question 117 | Q00526

Which of these is a super class of all errors and exceptions in the Java language?

A. Catchable

B. Throwable

C. RunTimeExceptions

D. None of the above

Hint

Separate source errors from runtime exceptions and determine the hierarchy and checked obligation.

Answer B: Throwable.

OOP29 places Exception and Error under the common root Throwable. Therefore Throwable is the superclass covering all errors and exceptions in the question. RuntimeException is only a sub-branch and cannot be the root of Error or checked exceptions. Catchable is not the root type named in the provided hierarchy.

Why each choice is right or wrong

A - Incorrect: Catchable is not the root class specified here’s throwable hierarchy.

B - Correct: Throwable is explicitly the shared root above both Exception and Error.

C - Incorrect: RuntimeException is an unchecked exception branch, not the common superclass of all errors and exceptions.

D - Incorrect: Throwable is offered and is the hierarchy root, so none of the above is false.

Primary note proof

Study-note section OOP29

> Throwable is the root of throwable values, with Exception and Error branches. RuntimeException and Error subclasses are unchecked; other throwable classes are checked.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 27 | lines 20-41

> • An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.
> • An exception is an indication of a problem that occurs during a program‟s execution.
> • An exception is a runtime error.
> • A Java exception is an instance of a class derived from Throwable.
> • The Throwable class is contained in the java.lang package, and subclasses of Throwable are contained in various packages.
> • Errors related to GUI components are included in the java.awt package; numeric exceptions are included in the java.lang package because they are related to the java.lang.Number class.
> • You can create your own exception classes by extending Throwable or a subclass of Throwable.

Object Oriented Programming module | PDF page 29 | lines 2-10

Exception Vs Errors
1. Errors: An Error indicates serious problem that a reasonable application should not try to catch.
2. Exception: Exception indicates conditions that a reasonable application might try to catch.

Question sources

Object-Oriented Programming practice questions · Question 96

Question 118 | Q00527

In which of the following package Exception class exist?

A. java.file

B. java.lang

C. java.io

D. java.util

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer B: java.lang.Exception belongs to java.lang.

The base Exception class belongs to java.lang, which is implicitly imported. Its place in the Throwable hierarchy does not determine the package of every subclass: I/O-specific subclasses may belong to java.io while the base remains java.lang.Exception. java.util contains utility types, and java.file is not the package of this class. Thus B supplies the correct package.

Why each choice is right or wrong

A - Incorrect: The official API places Exception in java.lang, not the proposed java.file package.

B - Correct: The official Java SE17 API explicitly identifies Exception as a java.lang class.

C - Incorrect: Specific I/O exceptions do not relocate the Exception base class from java.lang into java.io.

D - Incorrect: java.util provides utility types, while the authoritative API places Exception in java.lang.

Primary note proof

Study-note section OOP-S04

The base Exception class is java.lang.Exception. It does not belong to java.io merely because some subclasses represent I/O failures. Package membership and position in an inheritance hierarchy are separate facts.

Study-note section OOP29

Throwable is the root of throwable values, with Exception and Error branches.

Study-note section OOP28

java.lang is implicitly imported; java.util supplies utilities; java.io supplies I/O; java.awt includes Button and Checkbox.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 27 | lines 20-41

• An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.
• An exception is an indication of a problem that occurs during a program‟s execution.
• An exception is a runtime error.
• A Java exception is an instance of a class derived from Throwable.
• The Throwable class is contained in the java.lang package, and subclasses of Throwable are contained in various packages.
• Errors related to GUI components are included in the java.awt package; numeric exceptions are included in the java.lang package because they are related to the java.lang.Number class.
• You can create your own exception classes by extending Throwable or a subclass of Throwable.

Question sources

Object-Oriented Programming practice questions · Question 97

Question 119 | Q00528

Applets are designed to be embedded within an _____.

A. Javascript

B. Css

C. HTML

D. SQL

Hint

Use the historical host lifecycle and distinguish activation, suspension, drawing and unloading.

Answer C: An HTML document, in the historical applet model.

OOP34 explicitly says historical applet code was embedded in an HTML document and executed by a compatible JVM host/plugin or applet viewer. HTML is therefore the requested embedding format. The answer concerns legacy applets; it does not imply modern browsers execute applets or that JavaScript, CSS or SQL provides the described host document.

Why each choice is right or wrong

A - Incorrect: JavaScript is not the embedding document format identified for historical applets in OOP34.

B - Incorrect: CSS specifies styling rather than the HTML document into which the historical applet is embedded.

C - Correct: HTML is the document format explicitly named as the historical applet embedding context.

D - Incorrect: SQL is not the document format that hosts the historical applet in OOP34.

Primary note proof

Study-note section OOP34

> In the historical applet model, Java code was embedded in an HTML document and executed by a compatible JVM host/plugin or applet viewer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 8 | lines 32-36

> • Java classes do not need to have a
> main( ) method except the starting
> point for your program.
> • Further, applets don‟t require a main(
>) method at all.

Question sources

Object-Oriented Programming practice questions · Question 98

Question 120 | Q00529

Which of the following is required to view an applet?

A. JCM

B. JDM

C. JVM

D. Java class

Hint

Use the historical host lifecycle and distinguish activation, suspension, drawing and unloading.

Answer C: A JVM, together with a compatible historical host or applet viewer.

OOP34 says applets execute through a compatible JVM host/plugin or applet viewer. JVM is therefore the necessary component among the named options, although a JVM alone is not the entire browser embedding environment. This is a historical requirement rather than a statement that current browsers provide applet support.

Why each choice is right or wrong

A - Incorrect: JCM is not the JVM execution component specified by OOP34 for historical applet viewing.

B - Incorrect: JDM is not the JVM host or applet-viewer component identified here.

C - Correct: A compatible JVM is necessary to execute the applet, together with an appropriate historical host/viewer.

D - Incorrect: A Java class supplies code but does not itself provide the required JVM execution environment.

Primary note proof

Study-note section OOP34

> In the historical applet model, Java code was embedded in an HTML document and executed by a compatible JVM host/plugin or applet viewer.

Study-note section OOP34

> A JVM alone is also not the complete browser embedding environment.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 8 | lines 32-36

> • Java classes do not need to have a
> main( ) method except the starting
> point for your program.

• Further, applets don‟t require a main(
) method at all.

Object Oriented Programming module | PDF page 2 | lines 17-29

The Java compiler translates the program into an intermediate language called Java bytecodes.
Java bytecodes are platform-independent.
Guaranteed to be Write Once, Run Anywhere.
There are different Java interpreters for different platforms.
Java is: + Object Oriented + Platform Independent
+ Simple + Secure
+ Portable +
Multithreaded + Robust

Question sources

Object-Oriented Programming practice questions · Question 99

Question 121 | Q00530

Which method is automatically called after the browser calls the init method?

A. start

B. stop

C. destroy

D. paint

Hint

Use the historical host lifecycle and distinguish activation, suspension, drawing and unloading.

Answer A: start.

OOP34 says init initializes once and start follows initialization and resumes activity. Thus start is the lifecycle method called after init under the historical host model. stop suspends activity, destroy cleans up before unloading, and paint draws content; none states the next lifecycle stage after initialization.

Why each choice is right or wrong

A - Correct: start follows initialization and resumes applet activity, exactly matching the requested lifecycle point.

B - Incorrect: stop suspends activity when deactivated; it is not the post-init startup callback.

C - Incorrect: destroy performs cleanup before unloading, rather than starting activity immediately after initialization.

D - Incorrect: paint draws content and is not the lifecycle startup callback that follows init.

Primary note proof

Study-note section OOP34

> init initializes once; start follows initialization and resumes activity; stop suspends activity when the host deactivates it; destroy performs final cleanup before unloading. paint draws content and is not a constructor.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 8 | lines 32-36

> • Java classes do not need to have a
> main( ) method except the starting
> point for your program.
> • Further, applets don‟t require a main(
>) method at all.

Question sources

Object-Oriented Programming practice questions · Question 100

Question 122 | Q00531

In the historical Java applet lifecycle, which callback performs final cleanup when a compatible host normally unloads the applet?

A. start

B. stop

C. destroy

D. paint

Hint

Use the historical host lifecycle and distinguish activation, suspension, drawing and unloading.

Answer C: destroy

The applet lifecycle assigns final cleanup before unloading to destroy. start begins or resumes activity, stop handles host deactivation, and paint draws content. A normal browser shutdown may cause an unload, but unloading can also occur without shutting the browser down. The revised stem therefore asks for the cleanup callback directly and does not make an incorrect exclusive shutdown claim or promise cleanup after an abrupt crash.

Why each choice is right or wrong

A - Incorrect: start begins or resumes activity rather than performing final cleanup.

B - Incorrect: stop suspends activity on host deactivation, not final unloading.

C - Correct: destroy performs final cleanup before normal applet unloading.

D - Incorrect: paint draws content rather than handling final cleanup.

Primary note proof

Study-note section OOP34

> init initializes once; start follows initialization and resumes activity; stop suspends activity when the host deactivates it; destroy performs final cleanup before unloading. paint draws content and is not a constructor.

Study-note section OOP34

> destroy is not defined exclusively as “browser shuts down”: the host may destroy an applet on unloading, and abrupt process termination does not guarantee cleanup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 8 | lines 32-36

> • Java classes do not need to have a
> main( ) method except the starting
> point for your program.
> • Further, applets don‟t require a main(
>) method at all.

Question sources

Object-Oriented Programming practice questions · Question 101

Question 123 | Q00566

What are non-static Java fields declared directly in a class, outside methods, constructors and initializer-block local scopes, called?

A. Instance variables

B. Local variables

C. Class variables

D. Static variables

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer A: Instance variables

Fields belong to a class declaration, but non-static fields hold separate state for each object and are called instance variables. Static fields are class variables shared by instances, so merely being outside a method would not distinguish them. The stem now explicitly excludes static and block-local declarations. Local variables belong to method, constructor or block scopes rather than to each object's field state.

Why each choice is right or wrong

A - Correct: Non-static object fields are instance variables.

B - Incorrect: Locals belong to executable scopes rather than the class's object state.

C - Incorrect: Class variables are static fields, excluded by the question.

D - Incorrect: Static variables are shared class fields, excluded by the question.

Primary note proof

Study-note section OOP05

> The introductory three-way grouping is local, instance and class/static variables. Locals occur in methods, constructors or blocks; they must be definitely assigned before use and cannot have public/private/protected modifiers. Fields occur in a class outside method bodies; non-static fields belong to each instance, while static fields belong to the class. Declaration location alone does not distinguish instance and static fields.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 5 | lines 14-38

> B. Instance variables:
> • Are declared in a class, but
> outside a method, constructor or
> any block.
> • Have default values.
> • are variables which are bound to
> the object itself. Meaning=>
> • Every instance of that class
> (object) has it's own copy of that
> variable
> • Are created when an object is
> created with the use of the
> keyword 'new' and destroyed
> when the object is destroyed.
> • Can be declared in class level
> before or after use.
> • Access modifiers can be given
> for instance variables.
> • Are visible for all methods,

constructors and block in the
class.
E.g. class MyClass {
float aFloat; }
NB: By default, a member declared
within a class is an instance member.

Object Oriented Programming module | PDF page 5 | lines 39-61

C. Class/Static variables
• Are declared with the static keyword in a class, but outside a method, constructor or a block.
• There would only be one copy of each class variable per class, regardless of how many objects are created from it.
• Are rarely used other than being declared as constants. (public static final)
• Are stored in the static memory.
• Static variables are created when the program starts and destroyed when the program stops.
• Visibility is similar to instance variables. However, most static variables are declared public
• Default values are same as instance variables.
• Static variables can be accessed by calling with the class name ClassName.VariableName.

Question sources

Computer Science model exam · 2015 · Question 35

Question 124 | Q00571

Which one of the following sending a message to an object is correct?

A. obj.add

B. obj.add.obj

C. obj.add()

D. obj.add().obj

Hint

Separate parameters from arguments and operations that read state from operations that change it.

Answer C: obj.add()

An instance method is invoked with object.method(arguments), for example ob.test(). Sending the add message with no arguments follows obj.add().

Why each choice is right or wrong

A - Incorrect: Member selection without invocation parentheses

B - Incorrect: Does not form the demonstrated call syntax

C - Correct: Uses object receiver, method name and argument parentheses

D - Incorrect: Further .obj is not part of the demonstrated message invocation

Primary note proof

Study-note section OOP11

> An instance method can be invoked as object.method(), or with an implicit this inside suitable instance context.

Study-note section OOP01

> Objects send messages through method calls such as obj.add().

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 24-41

```
class OverloadDemo {
void test() {
System.out.println("No
parameters"); }
void test(int a) { // Overload
test for one integer parameter.
System.out.println("a: " + a); }
void test(int a, int b) { //
Overload test for two integer
parameters.
System.out.println("a and b: " +
a + " " + b); }
double test(double a) { //
overload test for a double
parameter
System.out.println("double a: "
+ a);
return a*a; } }
```

Object Oriented Programming module | PDF page 10 | lines 62-72

Methods
• Are members of classes and do
operations.
• Allows different classes to
communicate each other.
• E.g. type name(parameter-
list)
{ // body of method
//....
return value; // if
type is not void

Question sources

Computer Science model exam · 2015 · Question 40

Question 125 | Q00576

The process of binding both attributes and methods together within a class is?

A. Inheritance

B. Encapsulation

C. Polymorphism

D. Overriding

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer B: Encapsulation

Binding attributes and methods into one class is encapsulation.

Why each choice is right or wrong

A - Incorrect: Transfers properties across classes

B - Correct: Groups data and the operations on that data

C - Incorrect: Allows an operation to take different forms

D - Incorrect: Replaces an inherited method implementation

Primary note proof

Study-note section OOP02

Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface.

Study-note section OOP02

Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects. Prefer a precise mechanism over a vague slogan: wrapping fields and methods points to encapsulation; showing what an operation does while hiding how points to abstraction.

Study-note section OOP14

A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 35-45

ꣿ. Encapsulation:
• means the wrapping up of data and methods into a single unit (a class).
• It also means that the data inside a class is hidden from everything outside the class.
• The data can only be accessed by invoking the methods of the class.
• It is placing the data and the functions that work on that data in the same place.

Question sources

Computer Science model exam · 2015 · Question 45

Question 126 | Q00587

A keyword used to have only a single piece of storage for a particular field is?

A. this

B. object

C. Final

D. Static

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer D: Static

Only one copy of each class variable exists per class regardless of object count, and static is the declaration modifier for that property.

Why each choice is right or wrong

A - Incorrect: this does not provide shared storage

B - Incorrect: object does not declare the shared field category

C - Incorrect: final prevents modification rather than determining one copy

D - Correct: Static denotes the intended static modifier, though the literal Java spelling is lowercase static

Primary note proof

Study-note section OOP07

> static gives one field per loaded class, shared by its instances. If count starts at zero and every constructor increments it, three creations leave count=3. Prefer ClassName.count and ClassName.utility().

Study-note section OOP07

> final prevents reassignment of a variable, overriding of a method or extension of a class. A final reference may still refer to a mutable object.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 5 | lines 39-61

> C. Class/Static variables
> • Are declared with the static keyword in a class, but outside a method, constructor or a block.
> • There would only be one copy of each class variable per class, regardless of how many objects are created from it.
> • Are rarely used other than being declared as constants. (public static final)
> • Are stored in the static memory.
> • Static variables are created when the program starts and destroyed when the program stops.
> • Visibility is similar to instance variables. However, most static variables are declared public
> • Default values are same as instance variables.
> • Static variables can be accessed by calling with the class name ClassName.VariableName.

Question sources

Computer Science model exam · 2015 · Question 56

Question 127 | Q00589

What represents the concept that data can only be manipulated in known, well-defined ways

A. Inheritance

B. Information hiding

C. Polymorphism

D. overloading

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer B: Information hiding

Encapsulation hides class data and controls access through the class's methods. This is the information-hiding control over how data is manipulated.

Why each choice is right or wrong

A - Incorrect: Does not restrict manipulation paths

B - Correct: Describes control through the hidden-data/method boundary

C - Incorrect: Concerns multiple forms rather than restricting access

D - Incorrect: Concerns same-name methods with differing parameters

Primary note proof

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Study-note section OOP12

> Overloaded methods share a name but have different parameter types, counts or type order.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 35-45

ፍ. Encapsulation:
• means the wrapping up of data and methods into a single unit (a class).
• It also means that the data inside a class is hidden from everything outside the class.
• The data can only be accessed by invoking the methods of the class.
• It is placing the data and the functions that work on that data in the same place.

Question sources

Computer Science model exam · 2015 · Question 58

Question 128 | Q00599

Which one of the following is incorrect about inheritance

A. Subclass inherits members of superclass

B. Constructors are not members

C. Constructors of superclass can be invoked from subclass

D. Constructors of superclass inherited by subclass

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer D: Constructors of superclass inherited by subclass

Java constructors are not members and are not inherited, but a subclass can invoke its superclass constructor using super.

Why each choice is right or wrong

A - Incorrect: Subclass membership inheritance is the general mechanism

B - Incorrect: Constructors are not members.

C - Incorrect: Explicitly demonstrated with super(arg-list)

D - Correct: Contradicts the noninheritance rule

Primary note proof

Study-note section OOP-S05

In Java’s class-declaration terminology, constructors are not members and are not inherited. A subclass can invoke superclass construction; this does not turn the superclass constructor into an inherited subclass method.

Study-note section OOP10

Under Java 17 rules, an explicit this(args) or super(args) invocation must be the first constructor statement. this invokes another constructor in the same class; super invokes the immediate superclass constructor. A constructor cannot explicitly invoke both. If there is no explicit delegation, an implicit super() is inserted; a matching accessible no-argument parent constructor must exist.

Study-note section OOP13

class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 16 | lines 51-70

Types of inheritance
1. Simple/single inheritance: is damn easy to understand. When a class extends another one class only.
2. Multi-Level inheritance: One can inherit from a derived class, thereby making this derived class the base class for the new class.
3. Hierarchical inheritance: One class is inherited by many sub classes.
• Java does not support the inheritance of multiple super classes into a single subclass. (Multiple & Hybrid inheritances)
• No class can be a superclass of itself
• Superclass can be used by itself.
• The superclass has no knowledge of what a subclass adds to it.
• Constructors are not members, so they are not inherited by subclasses.

Object Oriented Programming module | PDF page 18 | lines 36-47

2. Used to access parent class constructor.
• A subclass can call a constructor defined by its superclass by use of the following form of super:
super(arg-list);
• Here, arg-list specifies any arguments needed by the constructor in the superclass.
• super( ) must always be the first

statement executed inside a subclass‟
constructor.

Question sources

Computer Science model exam · 2015 · Question 68

Question 129 | Q00605

Consider an abstract Java class that explicitly declares an abstract method. Which statement is incorrect about that class? In C, constructing objects means directly instantiating this abstract class.

A. It is a class containing abstract method

B. Abstract method is incomplete

C. Used to construct objects

D. None of the above

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer C: Used to construct objects

The given class contains an abstract method by assumption, and that method has no implementation body. An abstract class may include constructors and concrete methods, but it cannot itself be directly instantiated. C is therefore the false statement in the stated sense. A describes the given class rather than imposing a requirement on every abstract class, and None is false because C is incorrect.

Why each choice is right or wrong

A - Incorrect: This is true of the particular class because the stem explicitly says it declares an abstract method.

B - Incorrect: An abstract method has no implementation body, which is the intended meaning of incomplete.

C - Correct: Direct instantiation of the abstract class is prohibited, so this is the incorrect statement.

D - Incorrect: C is an incorrect statement, so None of the above cannot be selected.

Primary note proof

Study-note section OOP18

An abstract class cannot be directly instantiated but can define fields, constructors, abstract methods and concrete methods. It need not contain an abstract method. An abstract method has

no body; abstract and final conflict because an operation cannot require overriding while forbidding it.

Study-note section OOP18

A concrete subclass must provide or inherit implementations of all required abstract methods; otherwise it must itself be abstract.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 23 | lines 57-81

Abstract Classes
• A classes declared with abstract keyword and can‟t be instantiated.
• Used to hide meaningless methods:
• There can be no objects of an abstract class. .:.
• An abstract class cannot be directly instantiated with the new operator.
• You cannot declare abstract constructors, or abstract static methods
• Syntax: abstract class class_name
Example:
abstract class A {
abstract void callme();
// concrete methods are still allowed in abstract classes
void callmetoo() {
System.out.println("This is a concrete method.");
}}
class B extends A {
void callme() {
System.out.println("B's implementation of callme.");
}}

Object Oriented Programming module | PDF page 24 | lines 63-73

Summery:
Abstract:
• Classes: declared with abstract keyword and can‟t be instantiated.
• Methods: without body (no implementation)
Final:
• Class: declared as final prevents inheritance.

• Method: declared as final prevents overriding that method.

Question sources

Computer Science model exam · 2015 · Question 74

Question 130 | Q00608

Which statement correctly describes imports in an ordinary Java source compilation unit?

A. A Java source file may contain zero or more import statements before its top-level type declarations

B. Import statement appears after the class declaration

C. Import statement appears before package statement

D. All of the above

Hint

Check declaration order, exact case, package boundaries and the compile-versus-launch stage.

Answer A: A Java source file may contain zero or more import statements before its top-level type declarations

An optional package declaration comes first, followed by any import declarations, then the top-level type declarations. A source file can have no imports or multiple imports in that position. B places imports after a class, and C places them before the package declaration, reversing the ordering rules. D includes those false statements. These ordering rules apply to Java source files, rather than compiled .class contents.

Why each choice is right or wrong

A - Correct: This places zero or more source imports before the top-level types.

B - Incorrect: Imports precede top-level class declarations rather than following them.

C - Incorrect: The package declaration, when present, precedes imports.

D - Incorrect: B and C reverse the required ordering, so all is incorrect.

Primary note proof

Study-note section OOP28

package p; precedes imports; imports precede top-level type declarations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 51-70

2.3. Source file declaration
rules
• These rules are essential when declaring classes, import statements and package statements in a source file.
1. There can be only one public class per source file.
2. A source file can have multiple nonpublic classes.
3. The public class name should be the name of the source file as well which should be appended by .java at the end.
For example: the class name is public class Employee{} then the source file should be as Employee.java.
• If the class is defined inside a package, then the package statement should be the first statement in the source file.

Object Oriented Programming module | PDF page 10 | lines 2-12

package statement and the class declaration.
• If there are no package statements then the import statement should be the first line in the source file.
• Import and package statements will imply to all the classes present in the source file.
• It is not possible to declare different import and/or package statements to different classes in the source file.

Question sources

Computer Science model exam · 2015 · Question 77

Question 131 | Q00618

Which one of the following is correct about narrowing casting?

A. Done by the programmer

B. It is automatic

C. Done by the compiler

D. All

Hint

Identify the primitive category, width and conversion rule before evaluating a declaration or expression.

Answer A: Done by the programmer

Narrowing is explicit and must be done programmatically using a cast; the automatic conversion section instead describes widening.

Why each choice is right or wrong

A - Correct: Matches explicit programmatic narrowing

B - Incorrect: Confuses narrowing with widening

C - Incorrect: The compiler does not insert the narrowing cast described here

D - Incorrect: B and C do not match explicit narrowing

Primary note proof

Study-note section OOP06

> A compatible widening conversion often needs no cast; narrowing usually needs an explicit cast and may lose information. Widening an int to float can lose precision, so “widening never loses information” is false.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 42-55

> 2. Narrowing (Explicit)
> Casting: if you want to
> narrow or go in the other
> direction, Java won“t do that
> automatically. You have to
> do that programmatically.
> Syntax: smallerDT
> var=(smallerDT)largerDTVar
> OR largerValue;
> Example: float y=78.9f; byte
> b = (byte)y; //explicit type
> casting required
> int x = (int)2.7; double d =
> 100.04; long l = (long)d;

Question sources

Computer Science model exam · 2015 · Question 87

Question 132 | Q00626

Which one of the following is incorrect about method overriding?

A. A method declared final cannot be overridden

B. Similar argument list is required

C. Constructors cannot be overridden

D. None

Hint

Check inheritance, signature compatibility, visibility and eligibility for an override.

Answer D: None

final methods cannot be overridden, argument types and order must match, and constructors are not inherited methods. Hence none of A-C is incorrect under the intended exact-match reading.

Why each choice is right or wrong

A - Incorrect: Explicitly prohibited by overriding rules

B - Incorrect: Here, “similar argument list” must mean the same argument types in the same order.

C - Incorrect: Constructors are not inherited methods and have the class's own name

D - Correct: None of A-C contradicts these rules

Primary note proof

Study-note section OOP14

> A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype. Visibility cannot become more restrictive, and checked exceptions cannot be broadened incompatibly. @Override requests compiler checking; Java has no override keyword. Constructors, final methods, static methods and private parent methods are not overridden.

Study-note section OOP09

> Constructors are not inherited or overridden.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 21 | lines 26-39

> Rules of method overriding in Java..
> • Argument list: The argument list of
> overriding (method of subclass) and
> Overridden (method of super class)
> method must be similar.
> • The data types of the arguments and

their sequence should exactly match.
• Method names: must be similar.
• private, static and final methods cannot be overridden as they are local to the class.
• Access Modifier of the overriding method cannot be more restrictive than the overridden method of parent class.

Object Oriented Programming module | PDF page 16 | lines 51-70

Types of inheritance
1. Simple/single inheritance: is damn easy to understand. When a class extends another one class only.
2. Multi-Level inheritance: One can inherit from a derived class, thereby making this derived class the base class for the new class.
3. Hierarchical inheritance: One class is inherited by many sub classes.
• Java does not support the inheritance of multiple super classes into a single subclass. (Multiple & Hybrid inheritances)
• No class can be a superclass of itself
• Superclass can be used by itself.
• The superclass has no knowledge of what a subclass adds to it.
• Constructors are not members, so they are not inherited by subclasses.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often

sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Question sources

Computer Science model exam · 2015 · Question 95

Question 133 | Q00641

Which listed cases are not checked exceptions in Java? Interpret “array index out of bounds” as ArrayIndexOutOfBoundsException and “file required not found” as FileNotFoundException. Select all that apply.

A. Missing semicolons

B. Array index out of bounds

C. Usage of undeclared variables

D. File required not found

Hint

Separate source errors from runtime exceptions and determine the hierarchy and checked obligation.

Answer A, B, C: Missing semicolons; Array index out of bounds; Usage of undeclared variables

A missing semicolon and an undeclared variable are source errors rejected during compilation; neither is a checked exception. ArrayIndexOutOfBoundsException is a runtime exception and is unchecked. FileNotFoundException is a checked exception, so its catch-or-declare obligation is checked by the compiler even though an actual throw occurs at runtime. The revised multiple-selection question accepts all three cases outside the checked-exception category.

Why each choice is right or wrong

A - Correct: A missing semicolon is a compile-time source error, not a checked exception.

B - Correct: ArrayIndexOutOfBoundsException is unchecked.

C - Correct: An undeclared variable is a compile-time source error, not a checked exception.

D - Incorrect: FileNotFoundException is checked and therefore is not selected.

Primary note proof

Study-note section OOP29

IOException, FileNotFoundException and SQLException are standard checked examples. ArithmeticException, NullPointerException, ClassCastException and array bounds exceptions are unchecked. A missing semicolon or undeclared variable is a compile-time source error, not a checked exception.

Study-note section OOP29

Checked does not mean the exception occurs at compile time: the compiler checks an obligation to catch or declare it, while an actual throw occurs at runtime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 28 | lines 25-43

1. Checked exception: Is an exception that occurs at the compile time,
✓ These are also called as compile time exceptions.
✓ These exceptions cannot simply be ignored at the time of compilation,
✓ The Programmer should take care of (handle) these exceptions.
✓ For example, if you use FileReader class in your program to read data from a file, if the file specified in its constructor doesn't exist, then a FileNotFoundException occurs, and compiler prompts the programmer to handle the exception.

Object Oriented Programming module | PDF page 28 | lines 65-79

✓ Unchecked exceptions: Is an exception that occurs at the time of execution, these are also called as Runtime Exceptions.
✓ These include programming bugs, such as logic errors or improper use of an API.
✓ Runtime exceptions are ignored at the time of compilation.
✓ For example, if you have declared an

array of size 5 in your program, and trying to call the 6th element of the array then an ArrayIndexOutOfBoundsExceptione xception occurs.

Question sources

Computer Science model exam · 2015 · Question 110

Question 134 | Q00670

If a class is named Student, the class constructor name is_____..

A. Any legal java identifier

B. Any legal java identifier starting with s

C. Student

D. Student constructor

Hint

Check the class name, return-type absence and the constructors actually declared.

Answer C: Student

The constructor must have the same name as its class, so a Student constructor is Student.

Why each choice is right or wrong

A - Incorrect: An arbitrary identifier need not match the class

B - Incorrect: Starting with s is not sufficient and need not match Student

C - Correct: Exactly matches the class name

D - Incorrect: Adds an extra word not in the class name

Primary note proof

Study-note section OOP09

A constructor has the class's exact name and no return type, not even void.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately

upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Question sources

Computer Science model exam 2 · Question 11

Question 135 | Q00671

A visibility modifier that makes methods and data fields accessible only from within its own class.

A. Default

B. protected

C. public

D. private

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer D: private

The access table defines private as visible to the class only.

Why each choice is right or wrong

A - Incorrect: Default includes the package

B - Incorrect: Protected includes package and subclasses

C - Incorrect: Public is visible to the world

D - Correct: Matches class-only visibility

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether
> other classes can use a particular field or
> invoke a particular method.
> Their are four types access levels:
> 1. No Modifier(Default): Visible to
> the package(source file), and it is
> called package-private
> 2. Private: Visible to the class only.
> 3. Public: Visible to the world.
> 4. Protected: Visible to the package
> and all subclasses.

Question sources

Computer Science model exam 2 · Question 12

Question 136 | Q00672

The keyword used to create subclass/derived class in inheritance is_____

A. Implement

B. Implements

C. extend

D. extends

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer D: extends

The syntax is class Sub extends Super, using the exact lowercase keyword extends.

Why each choice is right or wrong

A - Incorrect: Not the class inheritance keyword

B - Incorrect: Not the class inheritance keyword

C - Incorrect: Missing the final s

D - Correct: Matches the stated extends spelling

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

> What is Inheritance?
> • Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
> • Inheritance is the process where one class acquires the properties (methods and fields) of another.
> • A class that is inherited is called a superclass.
> • The class that does the inheriting is called a subclass.
> • OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
> • The class whose properties are inherited is known as superclass (base class, parent class).
> • Therefore, a subclass is a specialized version of a superclass.
> • It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
> • The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
> • Sytax:
> class Super{
> .....}
> class Sub extends Super{ }

Question sources

Computer Science model exam 2 · Question 13

Question 137 | Q00673

Which of the following is not valid variable in java?

A. A$_2

B. By2

C. Java

D. A@2$

Hint

Check the lexical category and whether the notation names a value, a declaration or non-executed source text.

Answer D: A@2$ is invalid.

OOP28 explicitly allows $ in Java identifiers but rejects @ as an identifier character. D contains @, so it cannot be the requested variable name. A's dollar sign and underscore do not create that defect. By2 and Java also contain ordinary letters and a permitted trailing digit where present.

Why each choice is right or wrong

A - Incorrect: A$_2 uses a letter first and permitted dollar sign, underscore and digit thereafter.

B - Incorrect: By2 starts with a letter and has an allowed digit after the letters.

C - Incorrect: Java is a case-sensitive ordinary identifier, not the prohibited @ pattern identified in the note.

D - Correct: A@2$ contains @, which OOP28 explicitly prohibits within a Java identifier.

Primary note proof

Study-note section OOP-S01

> For ordinary ASCII Java identifiers, the first character is a letter, underscore or dollar sign; later characters may also be digits. A reserved keyword is not an ordinary identifier, and a lone underscore is not a valid identifier in Java 17.

Study-note section OOP28

> Java identifiers are case sensitive; $ is allowed but @ is not an identifier character.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 11-20

> ✓ Reserved keywords: class,
> extends, implements, for...

✓ Identifiers: programmers given name; grade, gRade... ✓ Literals (Constants values): 55, 60.78, “Hi”, „M“... Java language specifies five major types of literals => integer, floating, character, string & Boolean literals

Question sources

Computer Science model exam 2 · Question 14

Question 138 | Q00674

A blue print of an object is called _____.

A. operation

B. method

C. class

D. Attribute

Hint

Distinguish an object’s stored state, operations, identity and class definition.

Answer C: class

A class is the blueprint or template for an object.

Why each choice is right or wrong

A - Incorrect: Operation describes behavior

B - Incorrect: Method implements an operation

C - Correct: The stated blueprint concept

D - Incorrect: Attribute describes state

Primary note proof

Study-note section OOP01

An object has identity, state and behavior. State is stored in fields; behavior is provided by methods. A Rectangle object may hold width and height and answer getArea() or getPerimeter(). Two objects can have equal fields and still be distinct objects. A class defines their common structure and behavior; an object is a particular instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 2-18

• Are the basic runtime entities in an OO program.
• Are identifiable things or individuals?
• Are of significance to the system
• Have states reflected in the attributes (data) held about the object
• Have characteristic behaviour (i.e. functionality)
ꝺ. Class:
• Is a blueprint for an object.
• is the definition of a set of objects
• which are all similar in terms of their attributes
• A class consists of a class name, data and methods.

Object Oriented Programming module | PDF page 8 | lines 4-20

2.1. Fundamentals of Classes
• A class is a blue print from which individual objects are created.
• Class is a template for an object, and an object is an instance of a class
• Class is the logical construct upon which the entire Java language is built because it defines the shape and nature of an object.
• As such, the class forms the basis for object-oriented programming in Java.
• Classes usually consist of two things: instance variables and methods
• Any concept you wish to implement in a Java program must be encapsulated within a class.

Question sources

Computer Science model exam 2 · Question 15

Question 139 | Q00743

How is generalization implemented in Object-Oriented software engineering?

A. Inheritance

B. Polymorphism

C. Encapsulation

D. Abstract Classes

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer A: Inheritance

Generalization groups specialized classes under a general superclass; a subclass specializes its superclass, and inheritance implements that hierarchy.

Why each choice is right or wrong

A - Correct: Creates the generalized/specialized class hierarchy

B - Incorrect: Uses multiple forms within a hierarchy rather than creating its relationship

C - Incorrect: Groups data and methods

D - Incorrect: Can serve as a base but is not the general implementation mechanism required

Primary note proof

Study-note section OOP13

> Use inheritance when a subtype can satisfy the parent's contract: Dog is an Animal.

Study-note section OOP-S05

> Aggregation models one instance grouping other instances as a whole with parts. It is an object-relationship concept, not a method-dispatch rule. Inheritance instead relates a specialized class to a general class; overridden operations can then provide polymorphic behavior.

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

> What is Inheritance?
> • Inheritance is one of the cornerstones of
> object-oriented programming because it

allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Object Oriented Programming module | PDF page 3 | lines 46-56

5. Inheritance:
• It is the way in which objects of one class get the properties of objects of another class(data &methods).
• For example, Person, employee and retiree
• Inheritance provides the idea of reusability - the Person class can be reused without making changes to it, because another class can be made to inherit from it.

Question sources

Computer Science model exam 2 · Question 84

Question 140 | Q00747

Choose the incorrect statement in terms of Objects.

A. Objects are abstractions of real-world

B. Objects can't manage themselves

C. Objects encapsulate state and representation information

D. All of the mentioned

Hint

Distinguish an object’s stored state, operations, identity and class definition.

Answer B: “Objects cannot manage themselves” is the incorrect generalization.

OOP01 says objects hold state and behavior, control responsibilities, communicate through method calls and can delegate. It expressly calls the claim that objects cannot manage their own state misleading. Therefore B is the false generalization. A is a conventional modeling description, and C describes encapsulated state, so all statements cannot be incorrect.

Why each choice is right or wrong

A - Incorrect: Objects can model abstractions of real-world entities; this conventional modeling description is not the identified false generalization.

B - Correct: Objects can control their state through behavior and delegate responsibilities, contradicting this categorical claim.

C - Incorrect: Encapsulation of state and related representation is consistent with the note’s object model.

D - Incorrect: The supported abstraction and encapsulation descriptions prevent treating every supplied statement as incorrect.

Primary note proof

Study-note section OOP01

> An object has identity, state and behavior. State is stored in fields; behavior is provided by methods. A Rectangle object may hold width and height and answer getArea() or getPerimeter(). Two objects can have equal fields and still be distinct objects. A class defines their common structure and behavior; an object is a particular instance.

Study-note section OOP01

> Choose classes around responsibilities: an account controls its own balance, a transaction records a payment, and a service coordinates them. Objects send messages through method calls such as obj.add(). They can delegate work to other objects; saying that objects cannot manage their own state is misleading. Procedural and object oriented designs both support reuse; global shared data is not compulsory in procedural programming.

Study-note section OOP02

Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 2-18

• Are the basic runtime entities in
an OO program.
• Are identifiable things or
individuals?
• Are of significance to the system
• Have states reflected in the
attributes (data) held about
the object
• Have characteristic behaviour
(i.e. functionality)
፪. Class:
• Is a blueprint for an object.
• is the definition of a set of objects
• which are all similar in terms of their
attributes
• A class consists of a class name, data
and methods.

Object Oriented Programming module | PDF page 3 | lines 35-45

፫. Encapsulation:
• means the wrapping up of data and
methods into a single unit (a class).
• It also means that the data inside a
class is hidden from everything
outside the class.
• The data can only be accessed by
invoking the methods of the class.
• It is placing the data and the
functions that work on that data in
the same place.

Question sources

Computer Science model exam 2 · Question 88

Question 141 | Q00748

Which OOP concept describes one operation in a class hierarchy producing different behavior for different runtime object types?

A. Aggregation

B. Polymorphism

C. Inheritance

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer B: Polymorphism

Polymorphism lets code using a common type invoke behavior appropriate to each concrete object. Eligible overriding supplies this effect at runtime: the same inherited operation may dispatch to a different subclass implementation. Inheritance establishes the relationship that permits those related types, but does not alone name the varying behavior. Aggregation instead models a whole grouping parts, rather than method dispatch.

Why each choice is right or wrong

A - Incorrect: Aggregation describes whole-part object relationships, not varying operation behavior.

B - Correct: Polymorphism names behavior selected appropriately for different concrete object types through a common operation.

C - Incorrect: Inheritance establishes the subtype hierarchy; polymorphism names the differing behavior through that hierarchy.

Primary note proof

Study-note section OOP15

> For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Study-note section OOP-S05

> Aggregation models one instance grouping other instances as a whole with parts. It is an object-relationship concept, not a method-dispatch rule. Inheritance instead relates a specialized class to a general class; overridden operations can then provide polymorphic behavior.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 58-64

2. Polymorphism:
• The word polymorphism means the ability to take more than one form.
• In terms of the OOP, this means that a particular operation may behave differently for different sub-classes of the same class.

Object Oriented Programming module | PDF page 11 | lines 55-67

Method Overriding
In a class hierarchy, when a method in a subclass has the same name and type signature as a method in its superclass, this is called method Overriding.
• the method in the subclass is said to override the method in the superclass.
• The Method in the super class is known as an overridden method which is called within a subclass.
• The version of the method defined by the superclass will be hidden.

Question sources

Computer Science model exam 2 · Question 89

Question 142 | Q00896

Which statement correctly describes the implicit object argument in an ordinary C++17 member-function call?

A. The this pointer is supplied as the implicit object argument to ordinary non-static member functions

B. The this pointer is supplied as a hidden argument to static member variables

C. The this pointer is supplied as a hidden argument to static member functions

D. The this pointer is supplied as a hidden argument to every member function, including static functions

Hint

Use the stated language rules and identify whether a member call has an implicit current object.

Answer A: The this pointer is supplied as the implicit object argument to ordinary non-static member functions

In ordinary C++17 non-static member functions, this points to the object on which the function is invoked and can be understood as the implicit object argument. Member variables do not receive call arguments. Static member functions have no current instance and no this pointer, so a claim about every member function is also too broad. Correcting variables to functions in A restores the intended distinction.

Why each choice is right or wrong

A - Correct: Ordinary non-static member functions have the implicit object represented by this.

B - Incorrect: Variables are not calls and do not receive hidden object arguments.

C - Incorrect: Static member functions have no this pointer.

D - Incorrect: Including static functions makes the universal claim false.

Primary note proof

Study-note section OOP17

> In ordinary C++17 non-static member functions, this is a pointer to the object on which the function is called; this->field selects a member. It is often explained as an implicit object argument to a member function call. It is not an argument passed to member variables, and a static member function has no this pointer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 14 | lines 2-19

> Static methods and Final Variables
> • A static method can access only static data.
> • A static method can call only other static data (Static methods & variables), but can not call a non-static data from it.
> • Static method is belongs to the class and not to the object.
> • A static method can be accessed directly by the class name and doesn‟t need any object.
> • i.e ClassName.methodName(args)
> • A static method cannot refer to "this" or "super" keywords in anyway
> • But can be overloaded since they are resolved using static binding by compiler at compile time.

Question sources

Computer Science Exit Exam · 2018 · Question 9

Question 143 | Q00904

What does this Java program print?
public class Calculation {
public static void main(String[] args) {
int sum = 0;
for (int j = 1; j <= 10; j++) {
sum = sum + j;
}
System.out.println(+sum);
}
}

A. 55

B. 44

C. 33

D. 66

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer A: 55

The loop adds each integer from 1 through 10 exactly once. Their sum is 10×11/2=55, and unary plus does not change that integer before printing. The malformed trailing token is replaced by the required closing braces, leaving the loop itself unchanged. Forty-four and thirty-three are not the inclusive sum, while sixty-six would correspond to additionally summing 11, beyond the bound.

Why each choice is right or wrong

A - Correct: The inclusive sum 1+2+...+10 is 55.

B - Incorrect: 44 is not the sum of the ten loop values.

C - Incorrect: 33 is not the sum produced by the inclusive loop.

D - Incorrect: 66 would include an extra 11 beyond the stated upper bound.

Primary note proof

Study-note section OOP25

> A loop summing j=1..10 yields 55.

Study-note section OOP-S03

> Prefix ++ and -- update a variable before supplying its value; postfix forms supply the old value before the update. Multiplication, division and remainder bind before addition and subtraction;

same-level binary arithmetic groups left to right. Integer division truncates toward zero. Unary plus preserves its numeric operand’s value.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 7 | lines 26-41

Iteration: The for(), while() and
do...while() statements
4. Jumping Statements: to jump out
of loops and to control other areas of
program java uses break and
continue statements.
The break statement: used to exit a
loop
The Continue Statement: used to
force program to go back to the top
of a loop
Labels: Java does not include a goto
statement.
Instead of goto, Java allows you to
combine break and continue with a
label.

Object Oriented Programming module | PDF page 4 | lines 21-34

✓ Operators: Are symbols that
take one or more arguments
(operands) and operates on
them to produce a result. An
operator performs a function
on one (Unary), two (Binary),
or three operands (Ternary).
Java has one ternary operator,
?:, which is a short-hand if-
else statement. E.g. c=x?w:s;
Operators can be Arithmetic,
Logical, assignment,
Relational, conditional,
Increment,

Question sources

Computer Science Exit Exam · 2018 · Question 17

Question 144 | Q00905

Which exact case-sensitive Java keyword declares a subclass relationship to a superclass?

A. Inherits

B. Superclass

C. extends

D. Implements

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer C: extends

A Java class declares its superclass with extends, as in class Dog extends Animal. The lowercase spelling is required because Java identifiers and keywords are case sensitive. implements is used for interface contracts rather than the superclass relationship, while Inherits and Superclass are not the required keyword. Correcting the option's initial letter makes the subclass-declaration question exact without changing its concept.

Why each choice is right or wrong

A - Incorrect: Inherits is not the Java subclass-declaration keyword.

B - Incorrect: Superclass names the relationship but is not its declaration keyword.

C - Correct: extends is the exact keyword used between subclass and superclass names.

D - Incorrect: implements concerns interfaces, and the printed capitalized word is not the required keyword.

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Study-note section OOP19

> interface Payable{double getPaymentAmount();} declares an abstract public operation. class Invoice implements Payable must provide public double getPaymentAmount(), perhaps returning getQuantity()*getPricePerItem(). A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces.

Study-note section OOP28

> Java identifiers are case sensitive; $ is allowed but @ is not an identifier character.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Question sources

Computer Science Exit Exam · 2018 · Question 18

Question 145 | Q00930

A BankAccount keeps its balance and deposit operation together. Clients cannot assign balance directly and must use validated operations. Which principle specifically names this bundling of state with operations and control of access to that state?

A. Abstraction

B. Inheritance

C. Polymorphism

D. Encapsulation

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer D: Encapsulation controls which information of an object is exposed.

The question specifically asks for the mechanism that groups an object's state with its operations and controls access to that state. That is encapsulation. Abstraction also matters in the account's public view, but it names the essential operations presented to callers rather than the requested state-and-operation boundary. Inheritance and polymorphism concern subtype relationships and behavior through common types, respectively.

Why each choice is right or wrong

A - Incorrect: Abstraction emphasizes the essential caller-facing view; it does not specifically name the requested grouping and state-access boundary.

B - Incorrect: Inheritance forms a subtype relationship rather than the requested state-and-operation boundary.

C - Incorrect: Polymorphism selects behavior through a common type rather than grouping and controlling this state.

D - Correct: Encapsulation specifically groups the balance with its operations and controls direct access.

Primary note proof

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller. These ideas overlap: a private balance plus deposit(amount) can demonstrate all three, depending on what the question emphasizes.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 27-33

> 0. Abstraction:
> • provides only essential information
> to the outside world.
> • Or hiding their background details,
> i.e., to represent the needed
> information in program without
> presenting the details.

Object Oriented Programming module | PDF page 3 | lines 35-45

ვ. Encapsulation:
• means the wrapping up of data and methods into a single unit (a class).
• It also means that the data inside a class is hidden from everything outside the class.
• The data can only be accessed by invoking the methods of the class.
• It is placing the data and the functions that work on that data in the same place.

Question sources

Computer Science Exit Exam · 2018 · Question 43

Question 146 | Q00952

Consider the following code snippet
class Animal { void sound() { System.out.printIn("Animal sound"); } }
class Dog extends Animal { void sound() { System.out.printIn("Bark!"); } }
Which OOP concept is illustrated when Dog provides its own version of sound?

A. Method overriding

B. Method overloading

C. Abstraction

D. Encapsulation

Hint

Check inheritance, signature compatibility, visibility and eligibility for an override.

Answer A: Method overriding

Dog repeats sound() from Animal with its own body: that is a subclass method with the same name/signature as its superclass method. Literal printIn is a typo, but does not change the declaration relationship.

Why each choice is right or wrong

A - Correct: Matches the same-signature subclass relationship

B - Incorrect: No changed parameter list is shown

C - Incorrect: No abstract declaration or essential-interface example is needed

D - Incorrect: The example emphasizes replacement behavior, not grouping hidden data

Primary note proof

Study-note section OOP14

A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype.

Study-note section OOP12

Overloaded methods share a name but have different parameter types, counts or type order.

Study-note section OOP37

Check exact Java spelling and syntax: extends differs from Extends, println differs from printIn/printin, constructor names must match class names, and braces and operators such as <= and == affect validity and behavior. Java is case sensitive: an invalid program may not compile even when a possible trace seems clear.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 55-67

Method Overriding
In a class hierarchy, when a method in a subclass has the same name and type signature as a method in its superclass, this is called method Overriding.
• the method in the subclass is said to override the method in the superclass.
• The Method in the super class is known as an overridden method which is called within a subclass.
• The version of the method defined by the superclass will be hidden.

Question sources

Computer Science Exit Exam · 2018 · Question 65

Question 147 | Q00956

Which most restrictive Java member-access category admits unrelated classes in the same package and qualified subclass access in other packages, while excluding unrelated classes in other packages?

A. Public

B. Default

C. Private

D. Protected

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer D: Protected, subject to cross-package receiver restrictions.

Protected combines same-package access with qualified subclass access across packages. It therefore reaches both required contexts while excluding unrelated other-package clients. Public is broader and violates that exclusion. Package/default access does not reach the other-package subclass, while private is narrower than the required same-package access. Cross-package instance access must still use the inherited context or an appropriate subclass-typed receiver.

Why each choice is right or wrong

A - Incorrect: Public also admits unrelated classes in other packages, violating the explicit exclusion.

B - Incorrect: Package/default access does not admit the required subclass access across packages.

C - Incorrect: Private does not grant access to unrelated same-package classes.

D - Correct: Protected grants the required same-package and qualified cross-package subclass access without granting universal access.

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible. Top-level classes/interfaces may be public or package-private, not protected/private. Member types have additional visibility possibilities.

Study-note section OOP08

> Across packages, a subclass accessing a protected instance member must do so through its own inherited context or an appropriate subclass-typed receiver, not any arbitrary superclass object.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether
> other classes can use a particular field or
> invoke a particular method.
> Their are four types access levels:
> 1. No Modifier(Default): Visible to
> the package(source file), and it is
> called package-private
> 2. Private: Visible to the class only.
> 3. Public: Visible to the world.

4. Protected: Visible to the package
and all subclasses.

Question sources

Computer Science Exit Exam · 2018 · Question 69

Question 148 | Q00965

In abstract class Employee, calculateTax must remain a public concrete instance method while explicitly forbidding subclasses from overriding it. Which declaration achieves that requirement?

A. public abstract double calculateTax();

B. public final double calculateTax(){ return 0.0; }

C. private double calculateTax(){ return 0.0; }

D. public static double calculateTax(){ return 0.0; }

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer B: public final double calculateTax(){ return 0.0; }

final explicitly prohibits overriding an instance method while allowing it to remain public and concrete. abstract instead requires an implementation and supplies no body. private would remove public access and create independent child declarations rather than preserve the required public contract. static changes the method into a class operation whose same-signature child method hides it. Only the public final instance declaration satisfies all stated requirements.

Why each choice is right or wrong

A - Incorrect: abstract removes the concrete implementation and permits or requires overriding.

B - Correct: public final retains public concrete instance behavior and prohibits overriding.

C - Incorrect: private violates the required public accessibility.

D - Incorrect: static violates the requirement to remain an instance method.

Primary note proof

Study-note section OOP07

final prevents reassignment of a variable, overriding of a method or extension of a class.

Study-note section OOP14

A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype. Visibility cannot become more restrictive, and checked exceptions cannot be broadened incompatibly. @Override

requests compiler checking; Java has no override keyword. Constructors, final methods, static methods and private parent methods are not overridden.

Study-note section OOP18

An abstract method has no body; abstract and final conflict because an operation cannot require overriding while forbidding it.

Study-note section OOP08

For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP16

Fields are hidden rather than dynamically overridden, and same-signature static methods hide rather than override.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 24 | lines 6-16

Using final with Inheritance
• The keyword final has three uses.
1. To create the equivalent of a named constant (final variable)
- A variable declared as final prevents the content of that variable being modified.
2. To Prevent Overriding in Methods (final method) - A method declared as final cannot be overridden.

Object Oriented Programming module | PDF page 21 | lines 26-39

Rules of method overriding in Java..
• Argument list: The argument list of overriding (method of subclass) and Overridden (method of super class) method must be similar.
• The data types of the arguments and their sequence should exactly match.
• Method names: must be similar.
• private, static and final methods cannot be overridden as they are local to the class.
• Access Modifier of the overriding method cannot be more restrictive than the overridden method of parent class.

Question sources

Computer Science Exit Exam · 2018 · Question 78

Question 149 | Q00987

What will be the output for the Java expression: 8 - 2 * 5 * (3 + 9 / 3) + 6 / 3

A. -150

B. -30

C. -50

D. 50

Hint

Respect operator grouping and operand types, then trace each operation in evaluation order.

Answer C: -50.

Inside the parentheses, 9/3=3, so the group is 3+3=6. The multiplication gives 2*5*6=60, and the remaining integer division gives 6/3=2. The result is 8-60+2=-50, matching OOP23's explicit example. The negative sign follows from subtracting the larger product.

Why each choice is right or wrong

A - Incorrect: -150 does not follow the parenthesized value 6 and product 60 obtained from the stated operands.

B - Incorrect: -30 miscalculates the product or remaining terms; 8-60+2 is -50.

C - Correct: The grouped arithmetic gives 8-60+2, which equals -50.

D - Incorrect: The product is subtracted, so changing -50 to positive 50 loses the required sign.

Primary note proof

Study-note section OOP23

> In 8-2*5*(3+9/3)+6/3, the result is -50.

Study-note section OOP-S03

> Prefix ++ and -- update a variable before supplying its value; postfix forms supply the old value before the update. Multiplication, division and remainder bind before addition and subtraction; same-level binary arithmetic groups left to right. Integer division truncates toward zero. Unary plus preserves its numeric operand's value.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that
> take one or more arguments

(operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Question sources

National Exit Exam practice questions · Question 1

Computer Science Exit Exam practice collection · Question 80

Question 150 | Q00988

Which one of the following is used for constants in Java?

A. const

B. final

C. define

D. All of the above

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer B: final

final creates the equivalent of a named constant by preventing value modification.

Why each choice is right or wrong

A - Incorrect: Not the keyword given for constants

B - Correct: Explicitly stated named-constant mechanism

C - Incorrect: Not Java’s named-constant keyword

D - Incorrect: Only final is established

Primary note proof

Study-note section OOP07

final prevents reassignment of a variable, overriding of a method or extension of a class. A final reference may still refer to a mutable object. A shared mathematical PI is a good static final candidate; a student name normally belongs to an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 24 | lines 6-16

Using final with Inheritance
• The keyword final has three uses.
1. To create the equivalent of a named constant (final variable)
- A variable declared as final prevents the content of that variable being modified.
2. To Prevent Overriding in Methods (final method) - A method declared as final cannot be overridden.

Question sources

National Exit Exam practice questions · Question 2

Computer Science Exit Exam practice collection · Question 81

Question 151 | Q01172

In which character method returns the uppercase form of the specified char value.

A. a | toUpperCase()

B. i? | toLowerCase()

C. @) toString()

D. | isDigit()

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer A: Character.toUpperCase(ch).

OOP22 names Character.toUpperCase(ch) as the operation that converts a character to uppercase. A supplies the correct method name, with its argument abbreviated away. Lowercase conversion, string representation and a digit test have different purposes.

Why each choice is right or wrong

A - Correct: toUpperCase is the named conversion; an actual Character call supplies the char argument.

B - Incorrect: toLowerCase names the opposite letter-case conversion rather than producing uppercase.

C - Incorrect: toString names textual representation rather than uppercase conversion of the supplied character.

D - Incorrect: isDigit names a predicate about a character, not an uppercase conversion operation.

Primary note proof

Study-note section OOP-S04

> Character.toUpperCase(ch) supplies the uppercase equivalent where one exists; Character.toLowerCase(ch) supplies the lowercase equivalent. Character.isDigit(ch) tests whether a character is a digit and returns boolean. Character.toString(ch) supplies a one-character String rather than converting letter case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 14-16

> 5. String Class:- Immutable
> Class created to manipulate
> strings(Sequence of chars).

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)
> 2. Reference types: Arrays, classes,
> and interfaces, String)
> • Note: Boolean type has a value true or
> false (there is no conversion b/n Boolean
> and other types).
> • All non-primitive types are reference
> types, so all class types are reference
> types.
> • All integers in Java are signed, Java
> doesn‟t support unsigned integers.

Question sources

Computer Science Exit Exam practice questions · Question 33

Question 152 | Q01173

In which parameter is the regular expression to which this string is to be matched?

A. a | regul x

B. Q « x

C. |@| regex v

D. | None of the above x

Hint

Separate identity, content equality, immutable values and references that may be reassigned.

Answer C: regex, in the documented Java String.matches(String regex) signature.

In the Java String-matching API, matches(String regex) names its regular-expression parameter regex. This answers the API-signature question; it does not make regex a mandatory parameter name in arbitrary programmer-defined methods.

Why each choice is right or wrong

A - Incorrect: The image recovers regul, which is not the parameter name in OOP22's documented matches signature.

B - Incorrect: The image recovers reg; the documented String.matches parameter is named regex, not reg.

C - Correct: The image places regex at C, matching the regular-expression parameter named in OOP22.

D - Incorrect: The documented parameter name is present at C, so none of the above is false.

Primary note proof

Study-note section OOP22

> charAt(index) returns a char; length() counts UTF-16 units; substring(begin,end) excludes end. Character.toUpperCase(ch) converts a character, and matches(String regex) takes a regular-expression parameter named regex.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 14-16

> 5. String Class:- Immutable
> Class created to manipulate
> strings(Sequence of chars).

Question sources

Computer Science Exit Exam practice questions · Question 34

Question 153 | Q01212

Which one of the following is not the purpose of the super keyword?

A. Refer to the current object

B. Call superclass constructor

C. Access hidden variables

D. Invoke superclass method with the same name as the subclass method

Hint

Distinguish the current instance, parent context, hidden fields and static dispatch.

Answer A: Refer to the current object

super refers to the immediate superclass and can access its members or constructor. It is distinguished from this, which refers to the current instance.

Why each choice is right or wrong

A - Correct: Current-object role belongs to this, not the superclass qualifier

B - Incorrect: Explicit superclass-constructor use

C - Incorrect: super.member can resolve hidden variables

D - Incorrect: Explicit superclass-method access use

Primary note proof

Study-note section OOP16

> Java this denotes the current object; this.name=name distinguishes a field from a parameter. super.name selects a hidden superclass field; super.sound() directly requests the parent implementation. It does not create a second parent object. An unqualified instance call may dispatch dynamically; an explicit super call is different.

Study-note section OOP10

> Under Java 17 rules, an explicit this(args) or super(args) invocation must be the first constructor statement. this invokes another constructor in the same class; super invokes the immediate superclass constructor. A constructor cannot explicitly invoke both.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 17 | lines 51-74

> The super keyword
> • Since encapsulation is a primary
> attribute of OOP, Whenever a subclass
> needs to refer to its immediate
> superclass, it can do so by use of the
> keyword super.

• Super has two general forms.
1. Used to access parent class
instance members.
2. Used to access parent class
constructor.
1. Used to access parent class
instance members.
• super acts somewhat like this, except
that it always refers to the superclass of
the subclass in which it is used.
• Super has the form: super.member Here,
member can be either a method or an
instance variable.
• Super is most applicable to resolve
member hiding.
• super( ) must always be the first
statement executed inside a subclass‟
constructor.

Question sources

Computer Science model exam 2 · Question 33

Question 154 | Q01219

Which one of the following has the same name as the class name?

A. Method

B. Variable

C. Constructor

D. Package name

Hint

Check the class name, return-type absence and the constructors actually declared.

Answer C: Constructor

Constructors have the same name as their class and no return type; a regular method need not have that name.

Why each choice is right or wrong

A - Incorrect: No requirement for ordinary methods to match the class name

B - Incorrect: Variables have their own identifiers

C - Correct: This naming rule applies to constructors

D - Incorrect: Packages are organizational names, not constructor names

Primary note proof

Study-note section OOP09

A constructor has the class's exact name and no return type, not even void. It initializes the new object and can be overloaded or private.

Study-note section OOP28

Packages organize names and control access; hierarchical spelling does not grant special access between a package and its subpackages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Question sources

Computer Science model exam 2 · Question 40

Question 155 | Q01227

Which statement is true about the class concept?

A. The role of a class is to define the attributes and methods.

B. Class is used to distinguish one type of object from another.

C. A set of objects that share common methods, structure, and behavior.

D. All of the above

Hint

Distinguish an object’s stored state, operations, identity and class definition.

Answer D: All of the above

A class defines the common data/method shape of its objects and is itself a reference type. That supports attributes/methods, a distinct type, and a collection of similar objects.

Why each choice is right or wrong

A - Incorrect: Class consists of data and methods

B - Incorrect: Class defines a reference type and its objects' shape

C - Incorrect: Matches the definition of a set of similar objects

D - Correct: Combines the three supported descriptions

Primary note proof

Study-note section OOP01

> An object has identity, state and behavior. State is stored in fields; behavior is provided by methods. A Rectangle object may hold width and height and answer getArea() or getPerimeter(). Two objects can have equal fields and still be distinct objects. A class defines their common structure and behavior; an object is a particular instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 2-18

> • Are the basic runtime entities in
> an OO program.
> • Are identifiable things or
> individuals?
> • Are of significance to the system
> • Have states reflected in the
> attributes (data) held about
> the object
> • Have characteristic behaviour
> (i.e. functionality)
> ६. Class:

• Is a blueprint for an object.
• is the definition of a set of objects
• which are all similar in terms of their attributes
• A class consists of a class name, data and methods.

Object Oriented Programming module | PDF page 8 | lines 4-20

2.1. Fundamentals of Classes
• A class is a blue print from which individual objects are created.
• Class is a template for an object, and an object is an instance of a class
• Class is the logical construct upon which the entire Java language is built because it defines the shape and nature of an object.
• As such, the class forms the basis for object-oriented programming in Java.
• Classes usually consist of two things: instance variables and methods
• Any concept you wish to implement in a Java program must be encapsulated within a class.

Question sources

Computer Science model exam 2 · Question 48

Question 156 | Q01233

Which one is used to achieve the encapsulation concept in Java?

A. Declare the variables of a class as private.

B. Provide public setter and getter methods to modify and view the variable's values.

C. By using an Interface.

D. A & B

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer D: A & B

Hidden private fields plus accessible methods provide controlled access to class data. A alone hides data; B supplies the permitted interaction boundary.

Why each choice is right or wrong

A - Incorrect: Provides class-only data visibility

B - Incorrect: Provides the methods through which hidden data can be accessed

C - Incorrect: An interface alone does not hide a class's data fields

D - Correct: Combines hiding and controlled method access

Primary note proof

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface.

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP11

> An accessor/getter returns information; a mutator/setter changes state. Encapsulation does not require blindly generating setters for every field: validate changes and expose only useful operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 35-45

> ፭. Encapsulation:
> • means the wrapping up of data and methods into a single unit (a class).
> • It also means that the data inside a class is hidden from everything outside the class.
> • The data can only be accessed by invoking the methods of the class.
> • It is placing the data and the functions that work on that data in the same place.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether

other classes can use a particular field or invoke a particular method.
Their are four types access levels:
1. No Modifier(Default): Visible to the package(source file), and it is called package-private
2. Private: Visible to the class only.
3. Public: Visible to the world.
4. Protected: Visible to the package and all subclasses.

Question sources

Computer Science model exam 2 · Question 54

Question 157 | Q01234

Which one of the following is the collection of statements that performs a specific operation?

A. Class

B. Package

C. Variable

D. Method

Hint

Separate parameters from arguments and operations that read state from operations that change it.

Answer D: Method

Methods are class members that perform operations and contain executable bodies.

Why each choice is right or wrong

A - Incorrect: A class groups fields and methods, not just one operation's statements

B - Incorrect: Package groups classes/interfaces

C - Incorrect: Variable stores data rather than an operation body

D - Correct: Matches an operation implemented by a statement body

Primary note proof

Study-note section OOP11

A method has modifiers, return type, name, parameter list and body: public int getAge(){return age;}. A void method returns no value. Parameters name the inputs in the declaration; arguments supply values at a call.

Study-note section OOP01

An object has identity, state and behavior. State is stored in fields; behavior is provided by methods. A Rectangle object may hold width and height and answer getArea() or getPerimeter().

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 10 | lines 62-72

Methods
• Are members of classes and do
operations.
• Allows different classes to
communicate each other.
• E.g. type name(parameter-
list)
{ // body of method
//....
return value; // if
type is not void

Question sources

Computer Science model exam 2 · Question 55

Question 158 | Q01255

For this valid Java instance method inside a class, which input/output statement is correct?

```
public void test(boolean a, boolean b) {
if (a) {
System.out.println("A");
} else if (a && b) {
System.out.println("A && B");
} else {
if (!b) {
System.out.println("!B");
} else {
System.out.println("None");
}
}
}
```

A. If a and b are both true, the output is "A && B".

B. If a is true and b is false, the output is "!B".

C. If a is false and b is true, the output is "None".

D. If a and b are both false, the output is "None".

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer C: a=false and b=true prints None.

If a is true, the first branch prints A regardless of b. If a is false, the else-if condition a && b is necessarily false, so execution enters the final else. There !b selects !B when b is false and None when b is true. Only C pairs its input values with the resulting output.

Why each choice is right or wrong

A - Incorrect: When a is true the first branch prints A, not the unreachable A && B branch.

B - Incorrect: a=true selects A immediately regardless of b, so !B does not print.

C - Correct: a=false bypasses both earlier tests and b=true makes !b false, selecting None.

D - Incorrect: With both false, the final !b test is true and prints !B rather than None.

Primary note proof

Study-note section OOP25

> In if(a)... else if(a&&b)..., the second condition can never be true on a path reaching it because a is already false.

Study-note section OOP-S03

> For Boolean values, ! reverses true and false, && is true only when both operands are true, and || is true when at least one is true.

Study-note section OOP-S03

> An if selects its true branch or, when present, its else branch; only the selected branch executes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 7 | lines 48-54

> 2. Declaration statements: declares/
> Introduces a very first variables E.g.
> double c = 8.4;
> 3. Control flow statements: regulates
> the order of statements execution
> Selection: the if() and switch()
> statements

Object Oriented Programming module | PDF page 4 | lines 11-20

> ✓ Reserved keywords: class,
> extends, implements, for...
> ✓ Identifiers: programmers
> given name; grade, gRade...
> ✓ Literals (Constants values):

55, 60.78, “Hi”, „M“... Java language specifies five major types of literals => integer, floating, character, string & Boolean literals

Question sources

Computer Science model exam 2 · Question 76

Question 159 | Q01262

Which one of the following methods can be overriding?

A. Abstract method

B. Final Method

C. Private method

D. Static method

Hint

Check inheritance, signature compatibility, visibility and eligibility for an override.

Answer A: Abstract method

Subclasses provide implementations of inherited abstract methods; the overriding rules explicitly exclude private, static and final methods.

Why each choice is right or wrong

A - Correct: A child supplies the abstract method's implementation

B - Incorrect: Explicitly prohibited

C - Incorrect: Explicitly prohibited

D - Incorrect: Explicitly prohibited; static behavior is not overridden

Primary note proof

Study-note section OOP14

A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype.

Study-note section OOP14

Constructors, final methods, static methods and private parent methods are not overridden.

Study-note section OOP14

An abstract inherited method is implemented by an overriding method in a concrete subclass.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 21 | lines 26-39

Rules of method overriding in Java..
• Argument list: The argument list of overriding (method of subclass) and Overridden (method of super class) method must be similar.
• The data types of the arguments and their sequence should exactly match.
• Method names: must be similar.
• private, static and final methods cannot be overridden as they are local to the class.
• Access Modifier of the overriding method cannot be more restrictive than the overridden method of parent class.

Object Oriented Programming module | PDF page 23 | lines 57-81

Abstract Classes
• A classes declared with abstract keyword and can‟t be instantiated.
• Used to hide meaningless methods:
• There can be no objects of an abstract class. .:.
• An abstract class cannot be directly instantiated with the new operator.
• You cannot declare abstract constructors, or abstract static methods
• Syntax: abstract class class_name
Example:
abstract class A {
abstract void callme();
// concrete methods are still allowed in abstract classes
void callmetoo() {
System.out.println("This is a concrete method.");
}}
class B extends A {
void callme() {
System.out.println("B's implementation of callme.");
}}

Question sources

Computer Science model exam 2 · Question 83

Question 160 | Q01503

Considering the following program, Which one of the following statements gives syntax error?

```
class Shape {
double x;
void area() { ... }
}
class Circle extends Shape {
double PI = 3.14;
void display() { ... }
}
```

A. Shape s = new Circle(); s.area();

B. Shape s = new Shape(); s.area();

C. Shape s = new Circle(); s.display();

D. Circle c = new Circle(); c.area();

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer C: Shape s = new Circle(); s.display();

A superclass reference can refer to a child object, but only exposes members available through its declared type. An Animal reference cannot directly access the Dog-specific field breed. Shape has no display(), so s.display() is the analogous compile-time member error.

Why each choice is right or wrong

A - Incorrect: Shape reference can call its declared area; runtime dispatch applies

B - Incorrect: Shape object and its declared area are consistent

C - Correct: display is only declared in Circle, not the reference type Shape

D - Incorrect: Circle inherits area from Shape

Primary note proof

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 17 | lines 34-49

A Superclass Variable Can Reference a Subclass Object
class Animal{
string name; int age
int show(){...}}
class Dog extends Animal{
String breed;
String barking(){
}}
//main{
Animal wild=new Animal();
Dog fries=new Dog();
wild=fries;
Wild.show()// correct show() is defined in Animal
Wild.breed="Woo woo"// ERROR

Object Oriented Programming module | PDF page 22 | lines 37-53

• Dynamic method dispatch is the mechanism by which a call to an overridden method is resolved at run time, rather than compile time.
• Dynamic method dispatch is important because this is how Java implements run-time polymorphism
• When an overridden method is called through a superclass reference, Java determines which version of that method to execute based upon the type of the object being referred to at the time the call occurs.
• Thus, this determination is made at run time.
• Method Overriding is an example of runtime polymorphism.

Question sources

Mock exam 1 · Question 19

Question 161 | Q01504

One of the following does not describe an object

A. Delegates responsibilities to other objects

B. It receives and sends message from/to other objects

C. It is an instance of a class

D. It is an abstract description of the state and behavior of a collection of similar objects

Hint

Distinguish an object's stored state, operations, identity and class definition.

Answer D: It is an abstract description of the state and behavior of a collection of similar objects

An object is a concrete runtime instance with state and behavior, whereas the abstract description of similar objects is the class blueprint.

Why each choice is right or wrong

A - Incorrect: Can describe collaborating operations, though delegation is not separately defined

B - Incorrect: Method communication is consistent with the class operations description

C - Incorrect: Explicit object definition

D - Correct: Describes the class template rather than one instance

Primary note proof

Study-note section OOP01

> An object has identity, state and behavior. State is stored in fields; behavior is provided by methods. A Rectangle object may hold width and height and answer getArea() or getPerimeter(). Two objects can have equal fields and still be distinct objects. A class defines their common structure and behavior; an object is a particular instance.

Study-note section OOP01

> Objects send messages through method calls such as obj.add(). They can delegate work to other objects; saying that objects cannot manage their own state is misleading.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 2-18

> • Are the basic runtime entities in
> an OO program.
> • Are identifiable things or
> individuals?
> • Are of significance to the system
> • Have states reflected in the

attributes (data) held about
the object
• Have characteristic behaviour
(i.e. functionality)
ह. Class:
• Is a blueprint for an object.
• is the definition of a set of objects
• which are all similar in terms of their
attributes
• A class consists of a class name, data
and methods.

Object Oriented Programming module | PDF page 10 | lines 62-72

Methods
• Are members of classes and do
operations.
• Allows different classes to
communicate each other.
• E.g. type name(parameter-
list)
{ // body of method
//....
return value; // if
type is not void

Object Oriented Programming module | PDF page 8 | lines 4-20

2.1. Fundamentals of Classes
• A class is a blue print from which
individual objects are created.
• Class is a template for an object, and
an object is an instance of a class
• Class is the logical construct upon
which the entire Java language is
built because it defines the shape and
nature of an object.
• As such, the class forms the basis for
object-oriented programming in
Java.
• Classes usually consist of two things:
instance variables and methods
• Any concept you wish to implement
in a Java program must be
encapsulated within a class.

Question sources

Mock exam 1 · Question 20

Question 162 | Q01505

Which statements about Java 17 constructors are false? Select all that apply.

A. A constructor can be overloaded

B. It has the same name as the class

C. can be invoked within any method without object instantiation.

D. the first line of the constructor must be a call on another constructor in the same class

E. Has no return statement

Hint

Check the class name, return-type absence and the constructors actually declared.

Answer C, D, E: can be invoked within any method without object instantiation.; the first line of the constructor must be a call on another constructor in the same class; Has no return statement

Constructors can be overloaded and must use the class's exact name, so A and B are true. They are not ordinary methods that can be called anywhere without construction. A same-class this(...) call is not mandatory; delegation may use super(...) or implicit super(). Constructors also allow bare return; even though they have no return type and cannot return an expression. Therefore C, D and E are false.

Why each choice is right or wrong

A - Incorrect: Overloading constructors is permitted, so this statement is true.

B - Incorrect: A constructor has the exact class name, so this statement is true.

C - Correct: Constructors are not ordinary methods callable from any method without construction.

D - Correct: An explicit same-class constructor call is not mandatory.

E - Correct: A bare return; is allowed, so the claim of no return statement is false.

Primary note proof

Study-note section OOP09

> A constructor has the class's exact name and no return type, not even void. It initializes the new object and can be overloaded or private. A private constructor can support a factory or controlled singleton creation; it does not mean the class can never create an object internally.

Study-note section OOP09

> Constructors are not inherited or overridden. They may contain return; but cannot return an expression. Fields get their default values before ordinary constructor initialization, rather than receiving defaults only because a default constructor was generated.

Study-note section OOP10

Under Java 17 rules, an explicit this(args) or super(args) invocation must be the first constructor statement. this invokes another constructor in the same class; super invokes the immediate superclass constructor. A constructor cannot explicitly invoke both. If there is no explicit delegation, an implicit super() is inserted; a matching accessible no-argument parent constructor must exist.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Object Oriented Programming module | PDF page 18 | lines 36-47

2. Used to access parent class constructor.
• A subclass can call a constructor defined by its superclass by use of the following form of super:
super(arg-list);
• Here, arg-list specifies any arguments needed by the constructor in the superclass.
• super( ) must always be the first

statement executed inside a subclass‟
constructor.

Question sources

Mock exam 1 · Question 21

Question 163 | Q01585

What is the output of the following program? Assume every necessary part of the program is included.

```
class Car {
void drive() {
System.out.print("generic noise:");
}
}
public class Casting {
public static void main(String []args){
Car a = new Car();
Ferrari d = (Ferrari)a;
a.drive();
d.brake();
}
}
class Ferrari extends Car {
void drive() {
System.out.print("drive:");
}
void brake() {
System.out.println("brake hard:");
}
}
```

A. Generic noise: brake hard

B. drive: brake hard

C. generic nose: drive: brake hard

D. Compiler Error

E. Run time Error

Hint

Separate compile-time cast compatibility from the object’s actual runtime type.

Answer E: Runtime ClassCastException before any drive or brake output.

a refers to an actual Car created with new Car(), not to a Ferrari. The cast is type-checkable because Ferrari is a subclass, but it fails at runtime as OOP21 states. Execution stops at the cast with ClassCastException, before a.drive() or d.brake() can print any of their messages.

Why each choice is right or wrong

A - Incorrect: The failing cast occurs before either drive or brake, so neither listed message prints.

B - Incorrect: No Ferrari object exists here; the cast throws before any dynamic drive or brake call.

C - Incorrect: The program never reaches any of the three proposed messages because the cast fails first.

D - Incorrect: The related-class cast can pass compile-time checking; the actual object type causes a runtime failure.

E - Correct: Casting a plain Car object to Ferrari throws ClassCastException at runtime before any method output.

Primary note proof

Study-note section OOP21

> Upcasting a subclass reference to a superclass is implicit and does not change the object. A downcast may pass compile-time type checks yet fail at runtime. Car a=new Car(); Ferrari f=(Ferrari)a; throws ClassCastException before subsequent drive() or brake() calls. It does not turn a plain Car into a Ferrari.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 26 | lines 32-42

> The instance of operator
> • In Java, instanceof operator is used to
> check the type of an object at runtime.
> • It is the means by which your program
> can obtain run-time type information
> about an object.
> • instanceof operator is also important in
> case of casting object at runtime.
> • instanceof operator return boolean value,
> if an object reference is of specified type
> then it return true otherwise false.

Object Oriented Programming module | PDF page 26 | lines 48-69

> E.g. of downcasting with instanceof
> operator
> class Parent{ }
> public class Child extends

```
Parent{
public void check(){
System.out.println("Sucessfull
Casting");
}
public static void
show(Parent p){
if(p instanceof
Child){
Child
b1=(Child)p;
b1.check();
}}
public static void
main(String[] args){
Parent p=new Child();
Child.show(p);
}} Output: Sucessfull
```

Object Oriented Programming module | PDF page 17 | lines 34-49

```
A Superclass Variable Can Reference a
Subclass Object
class Animal{
string name; int age
int show(){...}}
class Dog extends Animal{
String breed;
String barking(){
}}
//main{
Animal wild=new Animal();
Dog fries=new Dog();
wild=fries;
Wild.show()// correct show() is
defined in Animal
Wild.breed="Woo woo"// ERROR
```

Question sources

Mock exam 3 · Question 17

Question 164 | Q01624

Which Java local declarations are valid when considered independently? Assume no type named x exists and the references are not read before assignment. Select all that apply.

A. None of the choices is correct

B. int [ ] x, y = new int[5];

C. int x[ ] = new x[5];

D. int [ ] x = new int[4];

E. int x = {5, 4, 9};

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer B, D: int [ ] x, y = new int[5];; int [ ] x = new int[4];

In B, int[] is the shared type of both x and y. Only y receives a new five-element array, which is a legal declaration as long as unassigned x is not read. D separately creates a four-element int array. C cannot use the variable-like name x as an allocation type under the stated scope, and E declares a scalar int with an array initializer. Thus B and D are valid.

Why each choice is right or wrong

A - Incorrect: B and D are valid declarations, so none is incorrect.

B - Correct: Both references have type int[]; initializing only y is legal.

C - Incorrect: No type x exists, so new x[5] is not a valid allocation here.

D - Correct: This is a valid int[] declaration and allocation.

E - Incorrect: A scalar int cannot be initialized with a multi-element brace array initializer.

Primary note proof

Study-note section OOP26

> int[] a=new int[5]; and int a[]={1,2,3}; are valid. int[] a={}; creates a zero-length array; explicit new is unnecessary in a declaration initializer. int a[],b; makes only a an array; int[] c,d; makes both arrays. Array objects have fixed length after creation, although the reference may be reassigned.

Study-note section OOP05

> Locals occur in methods, constructors or blocks; they must be definitely assigned before use and cannot have public/private/protected modifiers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Mock exam 2 · Question 13

Question 165 | Q01625

What is a Java static initializer block?

A. A block that initializes class-level state during class initialization, before that class's first relevant active use, including its main invocation

B. None of the choices is correct.

C. There is no such block.

D. It is used to create synchronized code.

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer A: A block that initializes class-level state during class initialization, before that class's first relevant active use, including its main invocation

A static initializer executes during class initialization and can initialize static state. It runs before the class's first relevant active use, including invoking its main method, rather than simply whenever a class file is loaded. It is a real Java construct, so none and the denial option are false. Synchronization is a different concern and does not define a static initializer's purpose.

Why each choice is right or wrong

A - Correct: This states the class-state initialization purpose and correct initialization timing.

B - Incorrect: A correctly describes a real construct, so none is incorrect.

C - Incorrect: Static initializer blocks exist in Java.

D - Incorrect: Creating synchronized code is not the defining role of a static initializer.

Primary note proof

Study-note section OOP07

A static initializer runs during class initialization, before the class's first relevant active use, not simply whenever a file is loaded.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 5 | lines 39-61

C. Class/Static variables
• Are declared with the static keyword in a class, but outside a method, constructor or a block.
• There would only be one copy of each class variable per class, regardless of how many objects are created from it.
• Are rarely used other than being declared as constants. (public static final)
• Are stored in the static memory.
• Static variables are created when the program starts and destroyed when the program stops.
• Visibility is similar to instance variables. However, most static variables are declared public
• Default values are same as instance variables.
• Static variables can be accessed by calling with the class name ClassName.VariableName.

Object Oriented Programming module | PDF page 14 | lines 2-19

Static methods and Final Variables
• A static method can access only static data.
• A static method can call only other static data (Static methods & variables), but can not call a non-static data from it.
• Static method is belongs to the class and not to the object.
• A static method can be accessed directly by the class name and doesn‟t need any object.

• i.e ClassName.methodName(args)
• A static method cannot refer to "this" or "super" keywords in anyway
• But can be overloaded since they are resolved using static binding by compiler at compile time.

Question sources

Mock exam 2 · Question 14

Question 166 | Q01626

What is the output of the below given Java code? Assume every necessary components of the program are included.

```
int i;
for (i = 0; i < 5; i++) {
if (i % 2 == 0)
System.out.print ( i + " ");
System.out.print ( i + 1);
}
```

A. 0 12 34 5

B. None of the choices is correct

C. 0 2 4 6

D. 0 122 344 5

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer D: 0 122 344 5.

Only the first print is controlled by if. At i=0,2,4 it prints the even value followed by a space; every iteration then prints i+1 without a separator. The contributions are "0 1", "2", "2 3", "4", "4 5". Joining them produces "0 122 344 5", exactly as OOP25 records.

Why each choice is right or wrong

A - Incorrect: This omits digits from the unconditional i+1 prints on odd and even iterations.

B - Incorrect: D supplies the exact concatenated output, so none of the choices is false.

C - Incorrect: The unconditional print runs five times; the output is not just an even-number sequence.

D - Correct: Joining every conditional and unconditional print yields the exact string 0 122 344 5.

Primary note proof

Study-note section OOP25

Count executions independently from printing. For i=0..4, if(i%2==0) print(i+" "); followed unconditionally by print(i+1); produces "0 122 344 5".

Study-note section OOP-S02

A print call emits its value without inserting a separator or ending the line. A println call emits its value followed by a line terminator. Spaces must come from the supplied text or explicit output operations; adjacent calls do not automatically separate their values.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 7 | lines 26-41

Iteration: The for(), while() and
do...while() statements
4. Jumping Statements: to jump out
of loops and to control other areas of
program java uses break and
continue statements.
The break statement: used to exit a
loop
The Continue Statement: used to
force program to go back to the top
of a loop
Labels: Java does not include a goto
statement.
Instead of goto, Java allows you to
combine break and continue with a
label.

Object Oriented Programming module | PDF page 4 | lines 21-34

✓ Operators: Are symbols that
take one or more arguments
(operands) and operates on
them to produce a result. An
operator performs a function
on one (Unary), two (Binary),
or three operands (Ternary).
Java has one ternary operator,
?:, which is a short-hand if-
else statement. E.g. c=x?w:s;
Operators can be Arithmetic,
Logical, assignment,
Relational, conditional,
Increment,

Question sources

Mock exam 2 · Question 15

Question 167 | Q00513B

_____ is a way of saying: This object is a type of that object.

A. IS-A

B. HAS-A

C. ARE-A

D. HAD-A

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer A: IS-A

IS-A describes type membership: an object belongs to a type or one of its subtypes.

Why each choice is right or wrong

A - Correct: Matches the named IS-A relationship

B - Incorrect: HAS-A would describe composition/containment rather than type membership

C - Incorrect: Not the named relationship

D - Incorrect: Not the named relationship

Primary note proof

Study-note section OOP13

> Use inheritance when a subtype can satisfy the parent's contract: Dog is an Animal. For shared Pet.name and speak(), put common state/contract in Pet, specialized fetch() in Dog, and appropriate speak() implementations in Dog and Cat. Composition represents has-a, such as Car containing Engine. Reuse alone does not justify an invalid is-a relationship.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 25 | lines 17-37

> • Polymorphism is the ability of an object
> to take on many forms.
> • The most common use of polymorphism
> in OOP occurs when a parent class
> reference is used to refer to a child class
> object.
> • Any Java object that can pass more than

one IS-A test is considered to be polymorphic.
• In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class Object.
• Polymorphism is derived from 2 Greek words: poly and morphs.
• The word "poly" means many and "morphs" means forms. So polymorphism means many forms.
• polymorphism is the capability of an action or method to do different things based on the object that it is acting upon.

Object Oriented Programming module | PDF page 30 | lines 37-59

Chapter - Five: Java Interface
• An interface in java is a blueprint of a class.
• It has static constants and abstract methods.
• The interface in java is a mechanism to achieve abstraction.
• There can be only abstract methods in the java interface not method body (contain only signature, not body).
• It is used to achieve abstraction and multiple inheritance in Java.
• Java Interface also represents IS-A relationship.
• It cannot be instantiated just like abstract class.
• There are mainly three reasons to use interface.
• It is used to achieve abstraction.
• By interface, we can support the functionality of multiple inheritance.

Question sources

Object-Oriented Programming practice questions · Question 83

Question 168 | SM0012

Which Java member access modifier permits subclasses in other packages to access inherited members through their own inherited context, while excluding unrelated clients in those other packages? Same-package access is allowed.

A. protected

B. private

C. default

D. public

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer A: protected

protected includes same-package access and also permits qualified subclass access across packages. That matches the revised boundary, which excludes unrelated clients in other packages while allowing same-package clients. private does not generally expose inherited members to separately declared subclasses, package access does not reach other packages, and public allows unrelated external clients. The question no longer demands an impossible universal subclasses-only rule.

Why each choice is right or wrong

A - Correct: protected provides same-package access plus the specified qualified cross-package subclass access.

B - Incorrect: private does not grant the requested general cross-package subclass access.

C - Incorrect: Package/default access does not extend to subclasses in another package.

D - Incorrect: public also permits unrelated accessible external clients, violating the boundary.

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP08

> Across packages, a subclass accessing a protected instance member must do so through its own inherited context or an appropriate subclass-typed receiver, not any arbitrary superclass object. protected therefore does not mean “subclasses only”: unrelated classes in the same package also have access.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

Access Specifiers
Access level modifiers determine whether other classes can use a particular field or invoke a particular method.
Their are four types access levels:
1. No Modifier(Default): Visible to the package(source file), and it is called package-private
2. Private: Visible to the class only.
3. Public: Visible to the world.
4. Protected: Visible to the package and all subclasses.

Question sources

MOE Computer Science Exit Exam · Question 3-4

National Computer Science Exit Exam · Question 63

National Computer Science Exit Exam · Question 66

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 97

Question 169 | SM0019

Which one of the following is the correct way of creating a String object?

A. String m = 'Computer Science';

B. String m = new String('Technology');

C. String m = 'H', 'e', 'l', 'l', 'o';

D. String m = new String("Exit Exam");

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer D: String m = new String("Exit Exam");

A String literal uses double quotes, while a character literal supplies one char in single quotes. D passes a valid String literal to the String constructor and assigns the resulting object reference. A and B instead use invalid multi-character character literals. C is not one valid String initializer: separate comma-delimited character literals do not form a String expression or a character-array constructor argument.

Why each choice is right or wrong

A - Incorrect: The multi-character text in single quotes is not a valid Java character literal or String expression.

B - Incorrect: new String does not make its invalid single-quoted multi-character argument legal.

C - Incorrect: The comma-separated character literals do not form one String initializer or a supplied char-array value.

D - Correct: A double-quoted String argument matches the String constructor and initializes the String reference.

Primary note proof

Study-note section OOP-S04

> A double-quoted string literal denotes a String object. new String(existingString) creates a String with the same text; new String(charArray) copies the array's characters into an immutable String. A single-quoted literal denotes one char, so several characters enclosed together in single quotes cannot supply a String.

Study-note section OOP03

> Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 14-16

> 5. String Class:- Immutable
> Class created to manipulate
> strings(Sequence of chars).

Object Oriented Programming module | PDF page 4 | lines 11-20

> ✓ Reserved keywords: class,
> extends, implements, for…
> ✓ Identifiers: programmers
> given name; grade, gRade…
> ✓ Literals (Constants values):
> 55, 60.78, "Hi", „M"… Java
> language specifies five major
> types of literals => integer,
> floating, character, string &
> Boolean literals

Object Oriented Programming module | PDF page 8 | lines 43-55

> • a class declaration only creates a
> template; it does not create an actual
> object.

• To actually create a Dog object, use the second statement.
• Dog myDog; // Creates only the template
• myDog = new Dog(); // create a Dog object called myDog
• After this statement executes, myDog will be an instance of Dog.

Question sources

MOE Computer Science Exit Exam · Question 5-3

National Computer Science Exit Exam · Question 68

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 98

Question 170 | SM0069

Which one of the following is false about String in Java?

A. In Java a string is treated as object

B. We can create a String object using String literal

C. We can create a String object using array of characters

D. Content of a string can be changed once the string is created

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer D: A created String's content cannot be changed.

OOP22 explicitly states that Java String objects are immutable. A variable can later reference a new String result, but the existing object's content does not change. D therefore contradicts the rule. The other choices describe String objects or ways of supplying initial content, which do not assert mutation of an already-created String.

Why each choice is right or wrong

A - Incorrect: String is an object type in the note; this is not the false mutability claim.

B - Incorrect: A String literal supplies String content; it does not imply the resulting object can later mutate.

C - Incorrect: Supplying initial character content creates a String; it does not change an existing String's immutable content.

D - Correct: This is false: Java String objects are immutable after creation, even if a variable is rebound.

Primary note proof

Study-note section OOP-S04

A double-quoted string literal denotes a String object. new String(existingString) creates a String with the same text; new String(charArray) copies the array's characters into an immutable String. A single-quoted literal denotes one char, so several characters enclosed together in single quotes cannot supply a String.

Study-note section OOP22

Java String objects are immutable. text += " World" assigns a reference to a new concatenated result and can print Hello World without mutating the original String object. final class String prevents subclassing; final alone does not make arbitrary objects immutable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 14-16

5. String Class:- Immutable
Class created to manipulate
strings(Sequence of chars).

Object Oriented Programming module | PDF page 8 | lines 43-55

• a class declaration only creates a
template; it does not create an actual
object.
• To actually create a Dog object, use
the second statement.
• Dog myDog; // Creates only
the template
• myDog = new Dog(); //
create a Dog object called
myDog
• After this statement executes,
myDog will be an instance of
Dog.

Object Oriented Programming module | PDF page 4 | lines 11-20

✓ Reserved keywords: class,
extends, implements, for…
✓ Identifiers: programmers
given name; grade, gRade…
✓ Literals (Constants values):

55, 60.78, “Hi”, „M“... Java language specifies five major types of literals => integer, floating, character, string & Boolean literals

Question sources

MOE Computer Science Exit Exam · Question 20-1

National Computer Science Exit Exam · Question 67

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 75

Question 171 | SM0074

Which concept refers to extending specialized classes from generalized classes in object-oriented programming?

A. Inheritance

B. Overriding

C. Overloading

D. Data field Encapsulation

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer A: Inheritance

A subclass is a specialized version of a superclass, and inheritance creates that relation.

Why each choice is right or wrong

A - Correct: Matches specialized-from-general hierarchy

B - Incorrect: Replaces an inherited method, not the class relationship

C - Incorrect: Adds same-name methods with changed parameters

D - Incorrect: Groups/controls data, not class specialization

Primary note proof

Study-note section OOP13

class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Study-note section OOP13

Use inheritance when a subtype can satisfy the parent's contract: Dog is an Animal.

Study-note section OOP02

Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Question sources

MOE Computer Science Exit Exam · Question 21-2

National Computer Science Exit Exam · Question 47

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 99

Question 172 | SM0218

One of the following is NOT TRUE about inheritance:

A. The subclass can add variables, but cannot redefine them

B. The derived class (subclass) is allowed to add methods or redefine them

C. A feature that allows us to derive a class from another class

D. Allows programmers to customize without actually modifying the original class (superclass)

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer A: The subclass can add variables, but cannot redefine them

B declares its own i, hiding A.i, and adds its own methods. Thus subclass fields can be declared with an ancestor's name; A's cannot claim is false in the intended hiding sense.

Why each choice is right or wrong

A - Correct: Contradicted by B's int i hiding A.i

B - Incorrect: Subclass can add methods and override inherited ones

C - Incorrect: Matches the inheritance definition

D - Incorrect: Subclass extension permits customization without changing the parent

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent. A class extends one direct class and can implement multiple interfaces. Single, multilevel and hierarchical inheritance mean one parent, a chain, and several children of one parent.

Study-note section OOP16

> Fields are hidden rather than dynamically overridden, and same-signature static methods hide rather than override.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 18 | lines 14-29

```
// Using super to overcome name
hiding.
class A {
int i; }
class B extends A {
int i; // this i hides the i in
A
B(int a, int b) {
super.i = a; // i in A
i = b; // i in B }
void show() {
System.out.println("i in
superclass: " + super.i);
System.out.println("i in
subclass: " + i);
}}
```

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{

```
.....}
class Sub extends Super{ }
```

Question sources

Computer Science Exit Exam · Sene 2016 · Question 6-2

Computer Science Exit Exam practice collection · Question 106

Question 173 | SM0237

Which OOP concept allows objects of different classes to be treated as objects of a common superclass?

A. Polymorphism

B. Inheritance

C. Abstraction

D. Encapsulation

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer A: Polymorphism

A parent-class reference referring to child objects is a common use of polymorphism. Inheritance establishes the relation, while polymorphism is the treatment through the common type.

Why each choice is right or wrong

A - Correct: Names the common-superclass many-forms treatment

B - Incorrect: Prerequisite relationship but not the treatment asked for

C - Incorrect: Hides background details

D - Incorrect: Groups data and methods

Primary note proof

Study-note section OOP15

> To solve dispatch questions: check the declaration and visibility, determine the compile-time signature, then identify the object's runtime class and its most specific override. An array Animal[]{new Dog(),new Cat()} can call speak() on each using one loop. New subclasses can supply behavior without rewriting this generic loop, which is the practical meaning of extensibility.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 25 | lines 17-37

• Polymorphism is the ability of an object to take on many forms.
• The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.
• Any Java object that can pass more than one IS-A test is considered to be polymorphic.
• In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class Object.
• Polymorphism is derived from 2 Greek words: poly and morphs.
• The word "poly" means many and "morphs" means forms. So polymorphism means many forms.
• polymorphism is the capability of an action or method to do different things based on the object that it is acting upon.

Object Oriented Programming module | PDF page 17 | lines 34-49

A Superclass Variable Can Reference a Subclass Object
class Animal{
string name; int age
int show(){...}}
class Dog extends Animal{
String breed;
String barking(){
}}
//main{
Animal wild=new Animal();
Dog fries=new Dog();
wild=fries;
Wild.show()// correct show() is defined in Animal
Wild.breed="Woo woo"// ERROR

Question sources

Computer Science Exit Exam · Sene 2016 · Question 12-2

Question 174 | SM0243

Which purposes apply to a Java interface viewed as a capability contract? “Blueprint” means an operation contract for implementing classes, and “multiple inheritance” means multiple type inheritance rather than multiple superclass state inheritance. Select all that apply.

A. To define a blueprint for a class

B. To allow multiple inheritance

C. To create objects directly

D. To hide internal implementation details

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer A, B, D: To define a blueprint for a class; To allow multiple inheritance; To hide internal implementation details

An interface specifies an operation contract that implementing classes satisfy, so the qualified blueprint description is valid. A class can implement multiple interfaces, gaining multiple type inheritance. Clients can use the contract without depending on a particular implementation’s internals, supporting abstraction. Interfaces are not directly instantiated, so creating objects directly is not their purpose. The multi-select form therefore accepts A, B and D.

Why each choice is right or wrong

A - Correct: An operation blueprint is the contract that implementations must satisfy.

B - Correct: Implementing multiple interfaces supports multiple type inheritance.

C - Incorrect: An interface cannot be instantiated directly.

D - Correct: The public contract lets callers use capabilities without depending on internal implementation details.

Primary note proof

Study-note section OOP19

> interface Payable{double getPaymentAmount();} declares an abstract public operation. class Invoice implements Payable must provide public double getPaymentAmount(), perhaps returning getQuantity()*getPricePerItem().

Study-note section OOP02

> Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Study-note section OOP20

> Use an abstract class for shared state, construction and common implementation; use an interface for a capability across unrelated classes and multiple type inheritance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 30 | lines 37-59

Chapter - Five: Java Interface
• An interface in java is a blueprint of a class.
• It has static constants and abstract methods.
• The interface in java is a mechanism to achieve abstraction.
• There can be only abstract methods in the java interface not method body (contain only signature, not body).
• It is used to achieve abstraction and multiple inheritance in Java.
• Java Interface also represents IS-A relationship.
• It cannot be instantiated just like abstract class.
• There are mainly three reasons to use interface.
• It is used to achieve abstraction.
• By interface, we can support the functionality of multiple inheritance.

Object Oriented Programming module | PDF page 32 | lines 12-22

Extending multiple Interface
Multiple inheritance in Java by interface
• If a class implements multiple interfaces, or an interface extends multiple interfaces i.e. known as multiple inheritance.
Extending multiple Interface
• A single interface can extend multiple interfaces, below is a simple example.

Variant 1 | Alternative version 1

Which purposes apply to Java interfaces as capability contracts, rather than complete per-instance state definitions? “Blueprint” means an operation contract, and “multiple inheritance” means multiple type inheritance. Select all that apply.

A. To create multiple constructors for a class

B. To define the state and behavior of an object

C. To define a blueprint of a class

D. To achieve multiple inheritance

Answer C, D: Interfaces define capability contracts and support multiple type inheritance.

This variant places the contract/blueprint purpose at C and multiple type inheritance at D. Both are valid: implementing objects satisfy the public operation contract, and a class may implement several interfaces. Interfaces have no constructors or per-instance fields, so they do not create class constructors or define an object’s complete state and implementation. Its selected labels therefore differ from the primary option set.

Why each choice is right or wrong

A - Incorrect: Interfaces have no constructors, so creating multiple constructors is not their purpose.

B - Incorrect: An interface specifies capabilities rather than complete per-instance state and implementation.

C - Correct: The defined blueprint is an operation contract for implementing classes.

D - Correct: Several interface types provide multiple type inheritance.

Primary note proof

Study-note section OOP19

> interface Payable{double getPaymentAmount();} declares an abstract public operation. class Invoice implements Payable must provide public double getPaymentAmount(), perhaps returning getQuantity()*getPricePerItem(). A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces. No interface instance fields or constructors exist; fields are implicitly public static final.

Study-note section OOP20

> Use an abstract class for shared state, construction and common implementation; use an interface for a capability across unrelated classes and multiple type inheritance.

Related reading (context, not primary note proof)

Study-note section OOP02

Question sources

Computer Science Exit Exam · Sene 2016 · Question 14-2

Computer Science MCQ practice collection · Question 340

Computer Science Exit Exam practice collection · Question 104

Question 175 | SM0245

Which OOP relationship is formed when a Java class implements several interface contracts? Choose the relationship concept, rather than the language construct declaring each contract.

A. Polymorphism

B. Encapsulation

C. Inheritance

D. Interface

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer C: Inheritance, specifically multiple interface/type inheritance.

Implementing several interfaces gives the class several interface types and forms multiple type inheritance. Inheritance is therefore the relationship concept requested. Interface names a construct used to declare a contract, rather than the relationship itself. Polymorphism can use the resulting common types to select behavior, while encapsulation groups state and controls access. Neither names the requested multiple-type relationship.

Why each choice is right or wrong

A - Incorrect: Polymorphism uses common types for appropriate behavior; the question asks for the type relationship itself.

B - Incorrect: Encapsulation groups state and controls access rather than forming the specified type relationship.

C - Correct: The class acquires multiple interface types, which is multiple type inheritance.

D - Incorrect: Interface is the construct defining a contract, not the requested relationship concept.

Primary note proof

Study-note section OOP19

> A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces.

Study-note section OOP20

> Use an abstract class for shared state, construction and common implementation; use an interface for a capability across unrelated classes and multiple type inheritance.

Study-note section OOP02

Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 32 | lines 12-22

Extending multiple Interface
Multiple inheritance in Java by interface
• If a class implements multiple interfaces, or an interface extends multiple interfaces i.e. known as multiple inheritance.
Extending multiple Interface
• A single interface can extend multiple interfaces, below is a simple example.

Object Oriented Programming module | PDF page 31 | lines 9-30

• It provides total abstraction;
• Means all the methods in interface are declared with empty body and are public and all fields are public, static and final by default.
• A class that implement interface must implement all the methods declared in the interface.
• If a class implements an interface and does not provide method bodies for all functions specified in the interface, then class must be declared abstract.
Rules of Interface
• Variables declared in an interface are public, static & final by default.
• Java does not allow you to extend more than one class, However you can implement more than one interfaces in your class.
• class implements interface but an interface extends another interfaces.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 15-1

Computer Science Exit Exam practice collection · Question 105

Question 176 | SM0283

Which of the following concepts in OOP allows a class to inherit attributes and methods from another class?

A. Inheritance

B. Abstraction

C. Encapsulation

D. Polymorphism

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer A: Inheritance

Acquiring another class's methods and fields defines inheritance.

Why each choice is right or wrong

A - Correct: Matches acquisition from another class

B - Incorrect: Essential information exposure

C - Incorrect: Wrapping data/methods together

D - Incorrect: Many forms of behavior, rather than the transfer relation

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

> What is Inheritance?
> • Inheritance is one of the cornerstones of

object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Question sources

Computer Science Exit Exam · Sene 2016 · Question 27-2

Computer Science Exit Exam practice collection · Question 103

Question 177 | SM0290

According to the introductory module's language-characteristics list, which description contradicts its characterization of Java?

A. Java is Object-Oriented

B. Java is Platform Independent

C. Java is difficult to learn

D. Java is architecture neutral

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer C: Java is difficult to learn

Java is commonly characterized as object oriented, platform independent and simple. Architecture-neutral bytecode is executed by a compatible platform-specific JVM. Difficult to learn conflicts with the module's intended simple characterization, making C the requested mismatch. This is a question about the module's description, not an objective claim that every learner finds Java easy or that every program runs unchanged in every environment.

Why each choice is right or wrong

A - Incorrect: Object orientation is part of the stated introductory characterization.

B - Incorrect: Platform independence is part of the stated characterization, subject to a compatible JVM and environment.

C - Correct: Difficult to learn conflicts with the module's intended simple characterization.

D - Incorrect: Architecture-neutral bytecode supports this characteristic within the specified compatible-runtime model.

Primary note proof

Study-note section OOP-S06

> Java is commonly characterized as object oriented, platform independent and simple. Its architecture-neutral bytecode is executed by a compatible platform-specific JVM. "Simple" is a teaching characterization relative to language design, not a promise that every learner finds every Java program easy.

Study-note section OOP39

> Java source normally compiles to JVM bytecode, and a compatible JVM plus libraries supports portability across systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

> The Java compiler translates the program
> into an intermediate language called Java
> bytecodes.
> Java bytecodes are platform-independent.
> Guaranteed to be Write Once, Run
> Anywhere.
> There are different Java interpreters for
> different platforms.
> Java is: + Object Oriented + Platform
> Independent

+ Simple + Secure
+ Portable +
Multithreaded + Robust

Question sources

Computer Science Exit Exam · Sene 2016 · Question 29-3

Computer Science Exit Exam practice collection · Question 95

Question 178 | SS094

Which statement about Java interfaces is false? In A, “interface instance” means an object made by directly instantiating the interface itself, without an implementation. C means interfaces may include abstract methods.

A. An interface can be instantiated directly without supplying an implementation, and its abstract methods can then be invoked

B. Like a class, an interface is a reference data type

C. Interface includes abstract methods

D. One class can implement multiple interfaces

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer A: An interface can be instantiated directly without supplying an implementation, and its abstract methods can then be invoked

An interface cannot be instantiated directly without an implementation. An interface reference can instead hold an implementing object and invoke its contract methods. Interfaces are reference types, can declare abstract operations, and can be implemented alongside other interfaces by one class. Clarifying A’s construction meaning isolates the false statement without incorrectly rejecting ordinary calls through interface-typed references.

Why each choice is right or wrong

A - Correct: Direct interface instantiation without an implementation is forbidden, so A is false.

B - Incorrect: An interface is a reference type usable for implementation references, so B is true.

C - Incorrect: An interface may declare abstract methods, so C is true.

D - Incorrect: One class can implement multiple interfaces, so D is true.

Primary note proof

Study-note section OOP19

interface Payable{double getPaymentAmount();} declares an abstract public operation. class Invoice implements Payable must provide public double getPaymentAmount(), perhaps

returning getQuantity()*getPricePerItem(). A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces.

Study-note section OOP19

Interfaces cannot be instantiated directly, but references may hold implementing objects, anonymous implementations or suitable functional-interface lambdas.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 30 | lines 37-59

Chapter - Five: Java Interface
• An interface in java is a blueprint of a class.
• It has static constants and abstract methods.
• The interface in java is a mechanism to achieve abstraction.
• There can be only abstract methods in the java interface not method body (contain only signature, not body).
• It is used to achieve abstraction and multiple inheritance in Java.
• Java Interface also represents IS-A relationship.
• It cannot be instantiated just like abstract class.
• There are mainly three reasons to use interface.
• It is used to achieve abstraction.
• By interface, we can support the functionality of multiple inheritance.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference

types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Object Oriented Programming module | PDF page 31 | lines 41-68

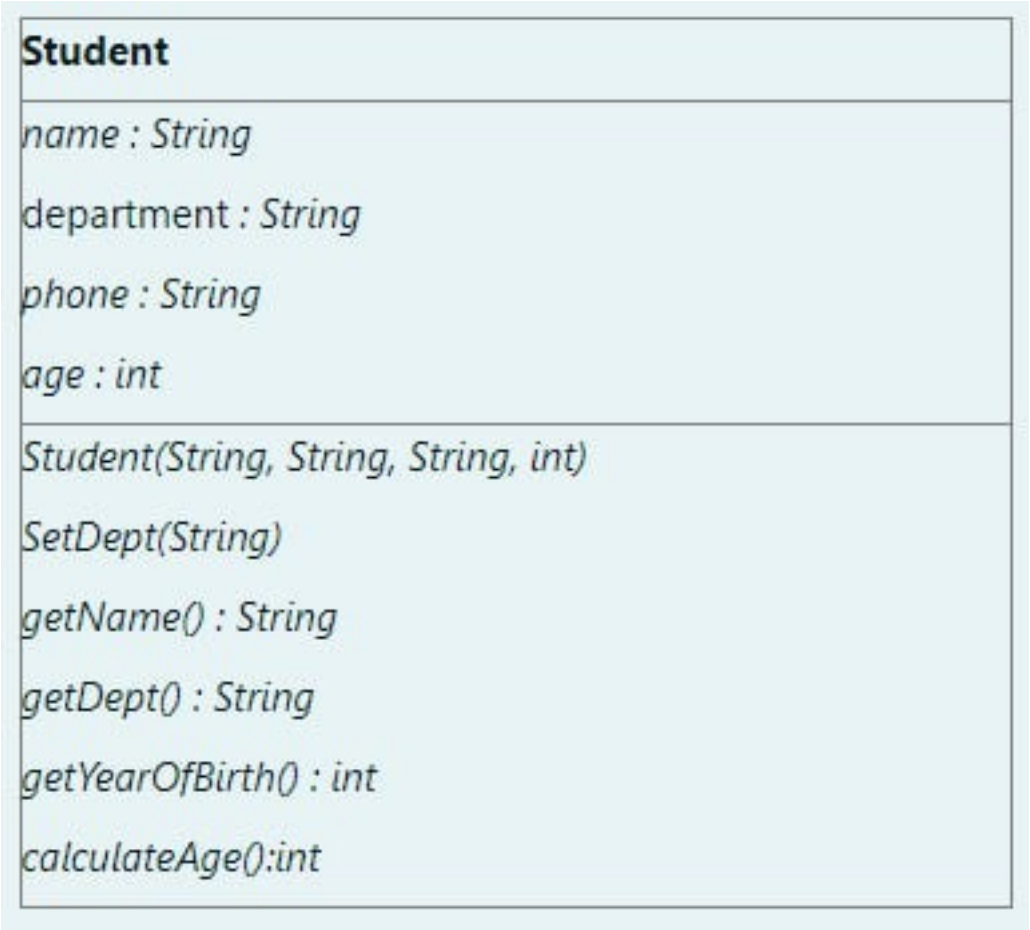
```
Example 2:
interface MyInterface{
/* compiler will treat them
as: public abstract void
method1(); public abstract void
method2(); */
public void method1();
public void method2(); }
class Demo implements
MyInterface{
/* This class must have to
implement both the abstract
methods else you will get
compilation error */
public void method1(){
System.out.println("imple
mentation of method1");
}
public void method2(){
System.out.println("imple
mentation of method2");
}
public static void
main(String arg[]){
MyInterface obj = new
Demo();
obj.method1();}}
Output:implementation of method1
```

Question sources

Computer Science model exam collection · Question 1

Question 179 | SS097

The following UML design shows Student class. Assume that the attributes have declared as private and the methods as public. The get methods
return the value of the attributes and the calculateAge() returns the Student age. Which of the following would not raise an error when you run this
program?



A. Student st1 = new Student ("John","CS","0913222222","25");

B. Student st1 = new Student ("John","SE","0913222222",20);

C. st1 Student = new Student ("John","CS","02085332455", 23);

D. Student st1 = new Student ("John","IT","2000");

Hint

Read the notation and distinguish a class description from a particular instance.

Answer B: Student st1 = new Student ("John","SE","0913222222",20);

The UML specifies Student parameters of types String, String, String and int, in that order. B supplies exactly those four argument types. Distinct String and int primitives/reference types rule out the quoted age in A.

Why each choice is right or wrong

A - Incorrect: Fourth argument is String 25 rather than int

B - Correct: Three String arguments and an int age match the UML

C - Incorrect: Reverses the required type/name declaration order

D - Incorrect: Only three String arguments; missing the required int age

Primary note proof

Study-note section OOP04

A UML class box normally contains name, attributes and operations. Visibility uses + public, - private and # protected. A field can be written name:String and an operation getName():String. A constructor Student(String,String,String,int) requires three strings and an integer, so an age argument "25" has the wrong type.

Study-note section OOP03

Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Object Oriented Programming module | PDF page 6 | lines 4-17

Java has two basic data types
1. Primitive Data Types: are 8 in java (byte, short, int, long, float, double, char & Boolean)
2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Object Oriented Programming module | PDF page 8 | lines 43-55

• a class declaration only creates a template; it does not create an actual object.
• To actually create a Dog object, use the second statement.
• Dog myDog; // Creates only the template
• myDog = new Dog(); // create a Dog object called myDog
• After this statement executes, myDog will be an instance of Dog.

Question sources

Computer Science model exam collection · Question 4

Question 180 | SS104

What we use to pass the data between activities in Android?

A. Broadcast receiver

B. PostgreSQL Database

C. Intent

D. Content Provider

Hint

Distinguish an interactive entry point, a requested action and attached data.

Answer C: An Intent, with extras.

OOP35 says an Intent describes an action or target and extras carry values between activities. Its putExtra("id",17) example supplies a concrete transfer. A broadcast receiver handles broadcasts and a content provider shares structured data; neither is the ordinary activity-start-with-parameters mechanism. A PostgreSQL database is not the inter-activity carrier described here.

Why each choice is right or wrong

A - Incorrect: A broadcast receiver handles broadcasts rather than serving as the ordinary activity-to-activity parameter carrier.

B - Incorrect: A database can store data but is not the Intent-and-extras transfer mechanism specified in OOP35.

C - Correct: An Intent selects the action/target and carries values between activities using extras.

D - Incorrect: A content provider shares structured data, rather than being the ordinary parameterized activity-start mechanism.

Primary note proof

Study-note section OOP35

> An Intent describes a requested action or explicit target; extras carry values between activities. For example, putExtra("id",17) attaches an ID for the destination to read. A content provider shares structured data, while a broadcast receiver handles broadcast messages; neither is the ordinary mechanism for starting an activity with parameters.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

> The Java compiler translates the program
> into an intermediate language called Java
> bytecodes.
> Java bytecodes are platform-independent.
> Guaranteed to be Write Once, Run
> Anywhere.
> There are different Java interpreters for
> different platforms.
> Java is: + Object Oriented + Platform
> Independent
> + Simple + Secure
> + Portable +
> Multithreaded + Robust

Question sources

Computer Science model exam collection · Question 11

Question 181 | SS108

What Is Manifest.xml in android?

A. It has the information about activities in an application

B. It has all the information about an application

C. Itis an executable file of the application

D. It has information about layout in an application

Hint

Separate application declarations, view containers and installation packaging.

Answer A: The manifest declares essential metadata, including application activities.

OOP36 says AndroidManifest.xml declares essential metadata such as components, permissions and intent filters. Activities are components, so A states a supported part of its contents. B overstates the scope by claiming all application information, which the note explicitly rejects. The image's reordered highlighted all-information choice cannot override that rule or the main option labels.

Why each choice is right or wrong

A - Correct: Activities are application components whose declarations form part of the manifest's essential metadata.

B - Incorrect: All information is too broad; the manifest does not contain all source code or layout details.

C - Incorrect: The manifest declares metadata and is explicitly not executable application code.

D - Incorrect: Layout resources describe views; the manifest does not contain the layout details asserted here.

Primary note proof

Study-note section OOP36

> AndroidManifest.xml declares essential app metadata such as components, permissions and intent filters. It is not executable code and does not contain all source or layout details.

Study-note section OOP36

> A manifest describes the package, while layout resources describe views.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

> The Java compiler translates the program
> into an intermediate language called Java
> bytecodes.
> Java bytecodes are platform-independent.

Guaranteed to be Write Once, Run Anywhere.
There are different Java interpreters for different platforms.
Java is: + Object Oriented + Platform Independent
+ Simple + Secure
+ Portable +
Multithreaded + Robust

Question sources

Computer Science model exam collection · Question 15

Question 182 | SS109

Assume you have an interface called Payable as follows:
public interface Payable {
double getPaymentAmount();
}
If you have another class Invoice that implements this interface and getQuantity() and getPricePerItem() methods that return quantity and price per item, which options correctly implement the Payable contract? Treat each option as an independent class skeleton with accessible getQuantity() and getPricePerItem() methods supplied and compatible with multiplication returning double. An abstract class may supply a concrete implementation. Select all that apply.

A. public abstract class Invoice implements Payable {
public double getPaymentAmount() {
return getQuantity() * getPricePerItem();
}
}

B. public class Invoice implements Payable {
public double getPaymentAmount() {
return getQuantity() * getPricePerItem();
}
}

C. public class Invoice implements Payable {
public getPaymentAmount() {
return getQuantity() * getPricePerItem();
}
}

D. public class Invoice extends Payable {
public double getPaymentAmount() {
return getQuantity() * getPricePerItem();
}
}

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

**Answer A, B: public abstract class Invoice implements Payable {
public double getPaymentAmount() {
return getQuantity() * getPricePerItem();
}
}; public class Invoice implements Payable {
public double getPaymentAmount() {
return getQuantity() * getPricePerItem();
}
}**

Both A and B provide public double getPaymentAmount() with the required quantity-times-unit-price result. Declaring A abstract does not prevent it from implementing the interface method concretely; it only prevents direct construction of Invoice itself. The assumed helper methods make both skeletons complete for this comparison. C omits the return type, while D incorrectly uses extends for the interface relationship. Thus A and B satisfy the contract.

Why each choice is right or wrong

A - Correct: An abstract class may still provide this concrete public double implementation.

B - Correct: The concrete class provides the required public double implementation.

C - Incorrect: The method lacks a return type and is not a valid implementation declaration.

D - Incorrect: A class implements Payable; it cannot extend that interface as a superclass.

Primary note proof

Study-note section OOP19

> interface Payable{double getPaymentAmount();} declares an abstract public operation. class Invoice implements Payable must provide public double getPaymentAmount(), perhaps returning getQuantity()*getPricePerItem(). A class can extend a superclass and implement zero or more interfaces.

Study-note section OOP18

> An abstract class cannot be directly instantiated but can define fields, constructors, abstract methods and concrete methods.

Study-note section OOP11

> A method has modifiers, return type, name, parameter list and body: public int getAge(){return age;}.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 31 | lines 9-30

> • It provides total abstraction;
> • Means all the methods in interface
> are declared with empty body and
> are public and all fields are public,

static and final by default.
• A class that implement interface must implement all the methods declared in the interface.
• If a class implements an interface and does not provide method bodies for all functions specified in the interface, then class must be declared abstract.
Rules of Interface
• Variables declared in an interface are public, static & final by default.
• Java does not allow you to extend more than one class, However you can implement more than one interfaces in your class.
• class implements interface but an interface extends another interfaces.

Object Oriented Programming module | PDF page 23 | lines 57-81

Abstract Classes
• A classes declared with abstract keyword and can‟t be instantiated.
• Used to hide meaningless methods:
• There can be no objects of an abstract class. .:.
• An abstract class cannot be directly instantiated with the new operator.
• You cannot declare abstract constructors, or abstract static methods
• Syntax: abstract class class_name
Example:

```
abstract class A {
abstract void callme();
// concrete methods are still
allowed in abstract classes
void callmetoo() {
System.out.println("This is a
concrete method.");
}}
class B extends A {
void callme() {
System.out.println("B's
implementation of callme.");
}}
```

Question sources

Computer Science model exam collection · Question 16

Computer Science model exam collection

Question 183 | SS117

Which statement is wrong about a Java try-catch construct? Assume any exception considered has a matching catch, and no earlier control transfer prevents reaching it.

A. The execution control is moved to catch block after exception occur in try block

B. When exception occurs, the execution of try block is interrupted

C. The catch block is executed once regardless of exceptions in try block

D. The catch block is used to handle exceptions that occur in try block

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer C: The catch block is executed once regardless of exceptions in try block

If the try block completes normally, its catch does not execute. If an exception occurs, normal execution of the try is interrupted and the matching handler runs. Thus the claim that catch runs once regardless of exceptions is false. Catch is conditional handling, whereas finally normally provides cleanup after the try and any selected handler. The matching-handler assumption makes the other transfer statements precise.

Why each choice is right or wrong

A - Incorrect: With the stated matching-handler assumption, an exception transfers control to that catch.

B - Incorrect: The exception interrupts normal execution and skips the remaining try body.

C - Correct: This is false: normal completion of try does not execute catch.

D - Incorrect: The matching catch handles the exception thrown during try.

Primary note proof

Study-note section OOP30

> try encloses guarded operations; a matching catch handles the thrown type or a subtype. Execution skips the remainder of the try when a throw occurs.

Study-note section OOP31

> finally normally executes after try and any selected catch, including when leaving by return or propagating an exception.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 28 | lines 8-22

```
try {
int num=
reader.nextInt();
System.out.println("The number
is :" + num);
}
catch
(InputMismatchException e){
System.out.println("There is
type mis match :");
}
finally{
System.out.println("This part is
always done");
}} }
```

Object Oriented Programming module | PDF page 29 | lines 11-22

```
Using try and catch...
General form of try/catch:
Try{
//block of code to monitor for
errors
}
Catch(ExceptionType1 exOb ) {
//handler for ExceptionType1
}
Catch(ExceptionType2 exOb ) {
//handler for ExceptionType2
}
```

Object Oriented Programming module | PDF page 27 | lines 20-41

```
• An exception is an event, which occurs
during the execution of a program, that
disrupts the normal flow of the
program's instructions.
• An exception is an indication of a
problem that occurs during a program‟s
execution.
• An exception is a runtime error.
• A Java exception is an instance of a class
derived from Throwable.
• The Throwable class is contained in the
java.lang package, and subclasses of
Throwable are contained in various
packages.
• Errors related to GUI components are
```

included in the java.awt package; numeric exceptions are included in the java.lang package because they are related to the java.lang.Number class.
• You can create your own exception classes by extending Throwable or a subclass of Throwable.

Question sources

Computer Science model exam collection · Question 24

Question 184 | SS118

An accessible abstract Class_A declares public abstract String welcome(); and has an accessible no-argument constructor. Which independent concrete definition of Class_B compiles? Each public class is in its own correctly named source file. These options restore the complete A-D code blocks from the source images.

A.
```
public class Class_B extends Class_A {
  public String whatObjectAmI() { return "I am a class B object"; }
  }
```

B.
```
public class Class_B extends Class_A {
  public int welcome() { int x = 10; return x; }
  }
```

C.
```
public class Class_B extends Class_A {
  public String whatObjectAmI() { return "I am a class D object"; }
  public String welcome() { return "I am a class C object"; }
  public int welcome(int y) { int x = 2 * y; return x; }
  }
```

D.
```
public class Class_B extends Class_A {
  public void welcome(String str) { }
  }
```

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer C: public class Class_B extends Class_A {
public String whatObjectAmI() { return "I am a class D object"; }
public String welcome() { return "I am a class C object"; }
public int welcome(int y) { int x = 2 * y; return x; }
}

The required inherited method has no parameters and returns String. C supplies that exact public implementation and may additionally overload welcome with an int parameter. A has no welcome implementation, B returns an incompatible int, and D supplies only a different-parameter overload. Those concrete subclasses leave the abstract contract unsatisfied or violate return compatibility.

Why each choice is right or wrong

A - Incorrect: whatObjectAmI() does not implement the required welcome() contract.

B - Incorrect: int is incompatible with the required String return for the same signature.

C - Correct: The exact public String welcome() implementation is present; welcome(int) is an allowed overload.

D - Incorrect: welcome(String) is an overload and does not implement no-argument welcome().

Primary note proof

Study-note section OOP18

> A concrete subclass must provide or inherit implementations of all required abstract methods; otherwise it must itself be abstract. If parent declares abstract String welcome(), a String welcome() implementation is required; int welcome() is incompatible, and welcome(int) merely adds an overload. A valid class can contain both the required implementation and an unrelated overload.

Study-note section OOP12

> Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 23 | lines 57-81

> Abstract Classes
> • A classes declared with abstract
> keyword and can‟t be instantiated.
> • Used to hide meaningless methods:
> • There can be no objects of an abstract
> class. .:.
> • An abstract class cannot be directly
> instantiated with the new operator.
> • You cannot declare abstract
> constructors, or abstract static methods

• Syntax: abstract class class_name
Example:
abstract class A {
abstract void callme();
// concrete methods are still
allowed in abstract classes
void callmetoo() {
System.out.println("This is a
concrete method.");
}}
class B extends A {
void callme() {
System.out.println("B's
implementation of callme.");
}}

Object Oriented Programming module | PDF page 11 | lines 55-67

Method Overriding
In a class hierarchy, when a method in a subclass has the same name and type signature as a method in its superclass, this is called method Overriding.
• the method in the subclass is said to override the method in the superclass.
• The Method in the super class is known as an overridden method which is called within a subclass.
• The version of the method defined by the superclass will be hidden.

Object Oriented Programming module | PDF page 11 | lines 7-18

Method Overloading
• Is a process of having the same name for different methods as long as their parameter list or data types in the parameter list is different?
• The return type alone is insufficient to distinguish two versions of a method.
• Parameters are key identifiers of OM.
• Is one of the ways that Java supports polymorphism

Question sources

Computer Science model exam collection · Question 25

Computer Science model exam collection

Question 185 | SS119

For traditional Android View-based layout containers, which class is their common container superclass?

A. android.view.Layout

B. android.view.RelativeLayout

C. android.view.ViewGroup

D. android.view.Widget

Hint

Separate application declarations, view containers and installation packaging.

Answer C: android.view.ViewGroup, for traditional view-based layout containers.

Traditional Android View-based layout containers, including LinearLayout and RelativeLayout, derive from android.view.ViewGroup. ViewGroup contains child views, matching the container role. RelativeLayout is one particular container rather than the common superclass, and the other supplied names do not identify that superclass. The View-system scope avoids extending this claim to every modern Android layout abstraction.

Why each choice is right or wrong

A - Incorrect: This is not the named View-system container superclass.

B - Incorrect: RelativeLayout is a particular layout container, not the common superclass of the layout containers.

C - Correct: android.view.ViewGroup is the child-view container superclass for this traditional View-system scope.

D - Incorrect: This does not name the View-system container superclass.

Primary note proof

Study-note section OOP36

> Traditional view-based layout containers such as LinearLayout and RelativeLayout derive from android.view.ViewGroup, which contains child views.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

The Java compiler translates the program into an intermediate language called Java bytecodes.
Java bytecodes are platform-independent.
Guaranteed to be Write Once, Run Anywhere.
There are different Java interpreters for different platforms.
Java is: + Object Oriented + Platform Independent
+ Simple + Secure
+ Portable +
Multithreaded + Robust

Question sources

Computer Science model exam collection · Question 26

Question 186 | SS121

Which statements about a Java abstract class are true? Select all that apply.

A. Cannot be instantiated directly

B. Must contain at least one abstract method

C. Can contain both abstract and non-abstract methods

D. Cannot be inherited

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer A, C: Cannot be instantiated directly; Can contain both abstract and non-abstract methods

An abstract class cannot be constructed directly, but it can contain both abstract and concrete methods. It can also be abstract without declaring any abstract method, so B is false. Abstract classes may be inherited; a concrete subclass must fulfill outstanding abstract method contracts. Cleaning the source marking noise and replacing the vague encapsulation slogan leaves two precise true properties at A and C.

Why each choice is right or wrong

A - Correct: Direct instantiation of an abstract class is prohibited.

B - Incorrect: An abstract class need not declare any abstract method.

C - Correct: Abstract and concrete methods may coexist in an abstract class.

D - Incorrect: Abstract classes can be inherited; concrete subclasses must fulfill their contracts.

Primary note proof

Study-note section OOP18

An abstract class cannot be directly instantiated but can define fields, constructors, abstract methods and concrete methods. It need not contain an abstract method. An abstract method has no body; abstract and final conflict because an operation cannot require overriding while forbidding it.

Study-note section OOP18

A concrete subclass must provide or inherit implementations of all required abstract methods; otherwise it must itself be abstract.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 23 | lines 57-81

Abstract Classes
• A classes declared with abstract
keyword and can‟t be instantiated.
• Used to hide meaningless methods:
• There can be no objects of an abstract
class. .:.
• An abstract class cannot be directly
instantiated with the new operator.
• You cannot declare abstract
constructors, or abstract static methods
• Syntax: abstract class class_name
Example:
abstract class A {
abstract void callme();
// concrete methods are still
allowed in abstract classes
void callmetoo() {
System.out.println("This is a
concrete method.");
}}
class B extends A {
void callme() {
System.out.println("B's
implementation of callme.");
}}

Question sources

Computer Science model exam collection · Question 28

Question 187 | SS129

What is an activity in android?

A. A single screen in an application with supporting java code

B. android class used to configure the android application

C. Itis an intent

D. android package file holding all the packages used

Hint

Distinguish an interactive entry point, a requested action and attached data.

Answer A: An Activity is a user-interaction entry component, often a screen.

An Android Activity is a component providing an entry point for user interaction, often corresponding to a screen. A is the introductory screen-and-code description that best matches this definition, though a one-to-one screen mapping is not universal. Configuration, Intents and package archives are separate concepts.

Why each choice is right or wrong

A - Correct: The screen with supporting code is the introductory description closest to the note's interaction-entry component.

B - Incorrect: Configuring the application describes a different responsibility than providing the Activity's user-interaction entry point.

C - Incorrect: An Intent describes a requested action or target; it is not itself an Activity.

D - Incorrect: An Android package archive is installation/runtime packaging, rather than a user-interaction component.

Primary note proof

Study-note section OOP35

> An Android Activity is a component providing an entry point for user interaction, often corresponding to a screen. An Intent describes a requested action or explicit target; extras carry values between activities. For example, putExtra("id",17) attaches an ID for the destination to read. A content provider shares structured data, while a broadcast receiver handles broadcast messages; neither is the ordinary mechanism for starting an activity with parameters.

Study-note section OOP36

> An .apk is an Android package archive used for installation and runtime contents.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

> The Java compiler translates the program
> into an intermediate language called Java

bytecodes.
Java bytecodes are platform-independent.
Guaranteed to be Write Once, Run
Anywhere.
There are different Java interpreters for
different platforms.
Java is: + Object Oriented + Platform
Independent
+ Simple + Secure
+ Portable +
Multithreaded + Robust

Question sources

Computer Science model exam collection · Question 36

Question 188 | SS140

Assume that there is a Pet family from which Pets like Dog and Cat share behavior from.Every Pet family has a name. However, the pet families have
some specific behaviors that are exhibited by only particular Pets. For instance, only Dogs can fetch something while both cats and dogs can speak.
Which of the following object oriented designprinciple would be suitable for this case?

A. Polymorphism

B. Encapsulation

C. Inheritance

D. _ Information hiding

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer C: Inheritance

A shared Pet superclass can hold name and common speak behavior; Dog and Cat specialize it, with fetch only in Dog. This directly uses inheritance. Overriding speak could also yield polymorphism but is not required by the stem.

Why each choice is right or wrong

A - Incorrect: Would apply if speak were dispatched to different overrides, which is not stated

B - Incorrect: Groups data/methods but does not establish shared family specialization

C - Correct: Models shared parent features plus subclass-specific behavior

D - Incorrect: Hides details rather than defining the family hierarchy

Primary note proof

Study-note section OOP13

Use inheritance when a subtype can satisfy the parent's contract: Dog is an Animal. For shared Pet.name and speak(), put common state/contract in Pet, specialized fetch() in Dog, and appropriate speak() implementations in Dog and Cat. Composition represents has-a, such as Car containing Engine. Reuse alone does not justify an invalid is-a relationship.

Study-note section OOP02

Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{

.....}
class Sub extends Super{ }

Object Oriented Programming module | PDF page 16 | lines 51-70

Types of inheritance
1. Simple/single inheritance: is damn easy to understand. When a class extends another one class only.
2. Multi-Level inheritance: One can inherit from a derived class, thereby making this derived class the base class for the new class.
3. Hierarchical inheritance: One class is inherited by many sub classes.
• Java does not support the inheritance of multiple super classes into a single subclass. (Multiple & Hybrid inheritances)
• No class can be a superclass of itself
• Superclass can be used by itself.
• The superclass has no knowledge of what a subclass adds to it.
• Constructors are not members, so they are not inherited by subclasses.

Object Oriented Programming module | PDF page 3 | lines 58-64

2. Polymorphism:
• The word polymorphism means the ability to take more than one form.
• In terms of the OOP, this means that a particular operation may behave differently for different sub-classes of the same class.

Question sources

Computer Science model exam collection · Question 47

Question 189 | SS165

Which method has the same name as that of its class?

A. constructor

B. class

C. delete

D. finalize

Hint

Check the class name, return-type absence and the constructors actually declared.

Answer A: constructor

The constructor naming rule requires the same name as its class. The term method is used informally in the question; constructors have no return type.

Why each choice is right or wrong

A - Correct: Matches the constructor naming rule

B - Incorrect: class is a declaration keyword, not this routine category

C - Incorrect: delete is not the constructor category

D - Incorrect: finalize has its own name rather than the class name

Primary note proof

Study-note section OOP09

> A constructor has the class's exact name and no return type, not even void.

Study-note section OOP-S05

> In Java's class-declaration terminology, constructors are not members and are not inherited.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 12 | lines 24-47

> Constructors
> • Performs automatic initialization of objects. .:.
> • Initializes an object immediately upon creation.
> • Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
> • The default constructor automatically initializes all instance variables to zero and all object references to NULL.
> • The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
> • Once you define your own constructor, the default constructor is no longer used.

• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Question sources

Computer Science model exam collection · Question 72

Question 190 | SS186

APK stands for:

A. Android Phone Kit

B. Android Page Kit

C. Android Platform Kit

D. Android Package Kit

Hint

Separate application declarations, view containers and installation packaging.

Answer D: Android Package Kit is the common exam expansion.

Android Package Kit is a common exam expansion of APK; the artifact is an Android package archive. Thus D is appropriate under this expansion convention. None of the Phone, Page or Platform alternatives describes this package archive.

Why each choice is right or wrong

A - Incorrect: Phone Kit does not match the common expansion or the package-archive meaning given in OOP36.

B - Incorrect: Page Kit does not describe the installation package archive identified here.

C - Incorrect: Platform Kit is not the bank expansion explicitly documented in OOP36.

D - Correct: Package Kit is the common exam expansion expressly stated, with Android package as the official artifact terminology.

Primary note proof

Study-note section OOP36

An .apk is an Android package archive used for installation and runtime contents. “Android Package Kit” is a common exam expansion; Android’s documentation calls it an Android package.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 2 | lines 17-29

The Java compiler translates the program into an intermediate language called Java bytecodes.
Java bytecodes are platform-independent.
Guaranteed to be Write Once, Run Anywhere.
There are different Java interpreters for different platforms.
Java is: + Object Oriented + Platform Independent
+ Simple + Secure
+ Portable +
Multithreaded + Robust

Question sources

Computer Science model exam collection · Question 93

Question 191 | AM0304

Which option specifically describes encapsulation’s bundling of related state and operations, rather than focusing on hiding implementation decisions behind an interface?

A. Combining data and methods into a single entity.

B. Hiding implementation details and providing a public interface.

C. Creating multiple instances of a class.

D. Inheriting properties and behavior from a base class.

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer A: Combining data and methods into a single entity.

Encapsulation groups related state and operations and controls access to them. A directly describes the grouping or bundling aspect requested here. B describes information hiding and can accompany encapsulation, but the stem deliberately asks for the grouping emphasis. Creating instances is object construction, while inheriting base properties is inheritance. This specific emphasis makes A the unique answer without denying the overlap of the broader principles.

Why each choice is right or wrong

A - Correct: Combining data and methods directly describes the requested bundling aspect.

B - Incorrect: This emphasizes information hiding rather than the specific grouping mechanism asked for.

C - Incorrect: Creating instances is object construction rather than encapsulation’s grouping.

D - Incorrect: Reusing base-class properties describes inheritance.

Primary note proof

Study-note section OOP02

Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller. These ideas overlap: a private balance plus deposit(amount) can demonstrate all three, depending on what the question emphasizes.

Study-note section OOP13

Use inheritance when a subtype can satisfy the parent's contract: Dog is an Animal.

Study-note section OOP03

Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 35-45

4. Encapsulation:
• means the wrapping up of data and methods into a single unit (a class).
• It also means that the data inside a class is hidden from everything outside the class.
• The data can only be accessed by invoking the methods of the class.
• It is placing the data and the functions that work on that data in the same place.

Object Oriented Programming module | PDF page 3 | lines 27-33

0. Abstraction:
• provides only essential information to the outside world.
• Or hiding their background details, i.e., to represent the needed information in program without presenting the details.

Question sources

Computer Science MCQ practice collection · Question 304

Question 192 | AM0305

What is inheritance in Java?

A. The process of creating objects from classes.

B. The ability of a class to inherit properties and behavior from another class.

C. Creating multiple instances of a class.

D. The ability to access variables and methods of a different class.

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer B: The ability of a class to inherit properties and behavior from another class.

Inheritance means acquiring another class's fields and methods, matching the properties-and-behavior description.

Why each choice is right or wrong

A - Incorrect: Instantiation rather than inheritance

B - Correct: Matches acquisition from a parent class

C - Incorrect: Instance creation rather than inheritance

D - Incorrect: Too broad: arbitrary access to another class is not inheritance

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Study-note section OOP03

> Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance. Member access uses s.name or s.study(), subject to visibility.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

> What is Inheritance?
> • Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
> • Inheritance is the process where one class acquires the properties (methods and fields) of another.
> • A class that is inherited is called a

superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Question sources

Computer Science MCQ practice collection · Question 305

Question 193 | AM0306

What is the concept of polymorphism in Java?

A. The ability to create multiple objects from a single class.

B. The ability to hide implementation details.

C. The ability to override methods in a subclass.

D. The ability of an object to take many forms and be treated as an instance of its superclass.

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer D: The ability of an object to take many forms and be treated as an instance of its superclass.

Polymorphism means many forms and can be illustrated by a parent reference to a child object. Overriding is one mechanism, not the full concept.

Why each choice is right or wrong

A - Incorrect: Creating several same-class objects does not imply many type forms

B - Incorrect: Hiding details describes abstraction

C - Incorrect: Overriding supports runtime polymorphism but is a narrower mechanism

D - Correct: Matches both many forms and common-superclass treatment

Primary note proof

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Study-note section OOP02

> Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 25 | lines 17-37

> • Polymorphism is the ability of an object to take on many forms.
> • The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.
> • Any Java object that can pass more than one IS-A test is considered to be polymorphic.
> • In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class Object.
> • Polymorphism is derived from 2 Greek words: poly and morphs.
> • The word "poly" means many and "morphs" means forms. So polymorphism means many forms.
> • polymorphism is the capability of an action or method to do different things based on the object that it is acting upon.

Object Oriented Programming module | PDF page 11 | lines 55-67

Method Overriding
In a class hierarchy, when a method in a subclass has the same name and type signature as a method in its superclass, this is called method Overriding.
• the method in the subclass is said to override the method in the superclass.
• The Method in the super class is known as an overridden method which is called within a subclass.
• The version of the method defined by the superclass will be hidden.

Object Oriented Programming module | PDF page 3 | lines 27-33

0. Abstraction:
• provides only essential information to the outside world.
• Or hiding their background details, i.e., to represent the needed information in program without presenting the details.

Question sources

Computer Science MCQ practice collection · Question 306

Question 194 | AM0307

What is the purpose of the super keyword in Java?

A. To access the superclass of a derived class.

B. To instantiate a new object.

C. To access the current object.

D. To define a method within a class.

Hint

Distinguish the current instance, parent context, hidden fields and static dispatch.

Answer A: To access the superclass of a derived class.

super selects the superclass context for member access and can request a superclass implementation. A hidden parent field can be selected with super.name, and super.sound() directly calls the parent version. It does not allocate a separate parent object or name the current instance; this is the current-object reference. Method definitions use declarations, so A is the valid purpose among the primary choices.

Why each choice is right or wrong

A - Correct: This describes selecting the superclass context and its members.

B - Incorrect: super does not allocate a separate object.

C - Incorrect: this denotes the current object; super selects parent context.

D - Incorrect: A method declaration defines a method; super does not do that.

Primary note proof

Study-note section OOP16

> Java this denotes the current object; this.name=name distinguishes a field from a parameter. super.name selects a hidden superclass field; super.sound() directly requests the parent implementation. It does not create a second parent object. An unqualified instance call may dispatch dynamically; an explicit super call is different.

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 17 | lines 51-74

> The super keyword
> • Since encapsulation is a primary attribute of OOP, Whenever a subclass needs to refer to its immediate superclass, it can do so by use of the keyword super.
> • Super has two general forms.
> 1. Used to access parent class instance members.
> 2. Used to access parent class constructor.
> 1. Used to access parent class instance members.
> • super acts somewhat like this, except that it always refers to the superclass of the subclass in which it is used.
> • Super has the form: super.member Here, member can be either a method or an instance variable.
> • Super is most applicable to resolve member hiding.
> • super( ) must always be the first statement executed inside a subclass‟ constructor.

Variant 1 | Alternative version 1

Inside an instance method declared in the current subclass, which descriptions correctly identify the parent-class context selected by Java super? Select all that apply.

A. It refers to the current object instance.

B. It refers to the superclass of the current class.

C. It refers to the class that contains the current method.

D. It refers to the class from which the current class is derived.

Answer B, D: Both B and D describe the superclass context selected by super.

The superclass of the current class and the class from which the current class derives name the same parent relationship, so B and D both apply. super selects that parent context for member access or explicit parent implementation calls. A instead describes this, while C identifies the current subclass containing the declared method. The variant therefore needs multiple selection even though the primary choices have one correct answer.

Why each choice is right or wrong

A - Incorrect: The current object instance is denoted by this, not the parent-context selector super.

B - Correct: The superclass is the parent context selected by super.

C - Incorrect: The method is declared in the current subclass, whereas super selects its parent context.

D - Correct: The class from which the current class derives is its superclass, equivalent to B.

Primary note proof

Study-note section OOP16

> Java this denotes the current object; this.name=name distinguishes a field from a parameter. super.name selects a hidden superclass field; super.sound() directly requests the parent implementation. It does not create a second parent object. An unqualified instance call may dispatch dynamically; an explicit super call is different.

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Question sources

Computer Science MCQ practice collection · Question 307

Computer Science MCQ practice collection · Question 332

Question 195 | AM0308

What is the difference between an abstract class and an interface in Java?

A. An abstract class can have concrete methods, while an interface cannot.

B. An interface can have static methods, while an abstract class cannot.

C. An abstract class can be instantiated, while an interface cannot.

D. An interface can extend multiple interfaces, while an abstract class cannot.

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer D: Interfaces can extend multiple interfaces; an abstract class instead implements interfaces and extends one class.

OOP19 permits an interface to extend several interfaces and a class to extend one superclass while implementing multiple interfaces. Thus D is correct when extend retains its Java declaration meaning. It must not be paraphrased as abstract classes cannot implement multiple interfaces. A is outdated for modern interfaces, B wrongly excludes class static methods, and C wrongly permits abstract instantiation.

Why each choice is right or wrong

A - Incorrect: Modern interfaces can contain default/static bodies, so concrete behavior is not exclusive to abstract classes.

B - Incorrect: Abstract classes can have static methods; their abstract modifier does not prohibit class-level methods.

C - Incorrect: An abstract class cannot be directly instantiated, so the first half of this distinction is false.

D - Correct: An interface extends multiple interfaces; a class implements them instead and has one direct superclass.

Primary note proof

Study-note section OOP19

> A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces.

Study-note section OOP19

> A method declaration without private, default or static is implicitly public abstract. Modern interfaces may also have default/static method bodies, and Java 9+ private helper methods.

Study-note section OOP18

> An abstract class cannot be directly instantiated but can define fields, constructors, abstract methods and concrete methods.

Study-note section OOP07

static gives one field per loaded class, shared by its instances. If count starts at zero and every constructor increments it, three creations leave count=3. Prefer ClassName.count and ClassName.utility().

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 23 | lines 57-81

Abstract Classes
• A classes declared with abstract keyword and can‟t be instantiated.
• Used to hide meaningless methods:
• There can be no objects of an abstract class. .:.
• An abstract class cannot be directly instantiated with the new operator.
• You cannot declare abstract constructors, or abstract static methods
• Syntax: abstract class class_name
Example:
abstract class A {
abstract void callme();
// concrete methods are still allowed in abstract classes
void callmetoo() {
System.out.println("This is a concrete method.");
}}
class B extends A {
void callme() {
System.out.println("B's implementation of callme.");
}}

Object Oriented Programming module | PDF page 30 | lines 37-59

Chapter - Five: Java Interface
• An interface in java is a blueprint of a class.
• It has static constants and abstract methods.
• The interface in java is a mechanism to achieve abstraction.
• There can be only abstract methods in the java interface not method body (contain only signature, not body).
• It is used to achieve abstraction and multiple inheritance in Java.

• Java Interface also represents IS-A relationship.
• It cannot be instantiated just like abstract class.
• There are mainly three reasons to use interface.
• It is used to achieve abstraction.
• By interface, we can support the functionality of multiple inheritance.

Object Oriented Programming module | PDF page 32 | lines 12-22

Extending multiple Interface
Multiple inheritance in Java by interface
• If a class implements multiple interfaces, or an interface extends multiple interfaces i.e. known as multiple inheritance.
Extending multiple Interface
• A single interface can extend multiple interfaces, below is a simple example.

Question sources

Computer Science MCQ practice collection · Question 308

Question 196 | AM0309

Which access modifier provides the widest accessibility in Java?

A. private

B. protected

C. public

D. default (no modifier)

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer C: public

World-visible public access is wider than class-only, package-only or package-plus-subclasses access.

Why each choice is right or wrong

A - Incorrect: Class-only

B - Incorrect: Package and subclasses

C - Correct: Visible to the world

D - Incorrect: Package-only

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether
> other classes can use a particular field or
> invoke a particular method.
> Their are four types access levels:
> 1. No Modifier(Default): Visible to
> the package(source file), and it is
> called package-private
> 2. Private: Visible to the class only.
> 3. Public: Visible to the world.
> 4. Protected: Visible to the package
> and all subclasses.

Question sources

Computer Science MCQ practice collection · Question 309

Question 197 | AM0310

What is the purpose of the this keyword in Java?

A. To refer to the superclass of a class.

B. To instantiate a new object.

C. To access the current object's variables and methods.

D. To define a method within a class.

Hint

Distinguish the current instance, parent context, hidden fields and static dispatch.

Answer C: Access the current object's variables and methods.

OOP16 says this denotes the current object and demonstrates this.name=name to distinguish a field from a parameter. That makes C the requested purpose. super selects parent context, new creates an object, and method declarations define methods. this neither creates a new instance nor denotes the superclass or the class itself.

Why each choice is right or wrong

A - Incorrect: Superclass context is selected using super, whereas this denotes the current object.

B - Incorrect: Object instantiation uses new and constructor invocation, not the this current-object reference.

C - Correct: this refers to the current object, enabling its instance fields and methods to be selected.

D - Incorrect: A method is defined by a declaration and body; this is an expression referring to the current object.

Primary note proof

Study-note section OOP16

> Java this denotes the current object; this.name=name distinguishes a field from a parameter. super.name selects a hidden superclass field; super.sound() directly requests the parent implementation. It does not create a second parent object.

Study-note section OOP03

> Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 17 | lines 51-74

> The super keyword
> • Since encapsulation is a primary attribute of OOP, Whenever a subclass needs to refer to its immediate superclass, it can do so by use of the keyword super.
> • Super has two general forms.
> 1. Used to access parent class instance members.
> 2. Used to access parent class constructor.
> 1. Used to access parent class instance members.
> • super acts somewhat like this, except that it always refers to the superclass of

the subclass in which it is used.
• Super has the form: super.member Here, member can be either a method or an instance variable.
• Super is most applicable to resolve member hiding.
• super( ) must always be the first statement executed inside a subclass‟ constructor.

Object Oriented Programming module | PDF page 18 | lines 14-29

```
// Using super to overcome name
hiding.
class A {
int i; }
class B extends A {
int i; // this i hides the i in
A
B(int a, int b) {
super.i = a; // i in A
i = b; // i in B }
void show() {
System.out.println("i in
superclass: " + super.i);
System.out.println("i in
subclass: " + i);
}}
```

Question sources

Computer Science MCQ practice collection · Question 310

Question 198 | AM0311

Which keyword is used to explicitly indicate that a method does not return any value in Java?

A. void

B. null

C. final

D. static

Hint

Separate parameters from arguments and operations that read state from operations that change it.

Answer A: void

The general method pattern returns a value only if type is not void, and test() is shown as void without a value return.

Why each choice is right or wrong

A - Correct: Explicit no-value-return type

B - Incorrect: null is not the no-value method return type shown

C - Incorrect: final prohibits modification/overriding, not returning a value

D - Incorrect: static belongs to the class, not a no-return indicator

Primary note proof

Study-note section OOP11

> A method has modifiers, return type, name, parameter list and body: public int getAge(){return age;}. A void method returns no value.

Study-note section OOP07

> final prevents reassignment of a variable, overriding of a method or extension of a class.

Study-note section OOP07

> static gives one field per loaded class, shared by its instances.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 10 | lines 62-72

> Methods
> • Are members of classes and do
> operations.
> • Allows different classes to
> communicate each other.
> • E.g. type name(parameter-
> list)
> { // body of method
> //....
> return value; // if
> type is not void

Object Oriented Programming module | PDF page 11 | lines 24-41

> class OverloadDemo {
> void test() {
> System.out.println("No
> parameters"); }
> void test(int a) { // Overload
> test for one integer parameter.

```
System.out.println("a: " + a); }
void test(int a, int b) { //
Overload test for two integer
parameters.
System.out.println("a and b: " +
a + " " + b); }
double test(double a) { //
overload test for a double
parameter
System.out.println("double a: "
+ a);
return a*a; } }
```

Question sources

Computer Science MCQ practice collection · Question 311

Question 199 | AM0312

What is method overloading in Java?

A. Defining multiple methods with the same name but different parameters in the same class.

B. Defining a method within another method.

C. Invoking a method from within itself.

D. Creating multiple instances of a class.

Hint

Compare names and parameter signatures before considering return types or access modifiers.

Answer A: Defining multiple methods with the same name but different parameters in the same class.

The definition retains one method name while differentiating parameter lists. Return type alone is insufficient, so signature differences must be in parameters.

Why each choice is right or wrong

A - Correct: Matches same name and different parameters

B - Incorrect: Nested method definition is not overloading

C - Incorrect: Self-invocation is recursion

D - Incorrect: Multiple objects are instantiation, not overloaded methods

Primary note proof

Study-note section OOP12

Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++. Overloads can also arise through inherited methods; “same class only” is a teaching simplification. f(int), f(double) and f(int,int) are distinct, but changing a parameter name does not create an overload.

Study-note section OOP03

Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 7-18

Method Overloading
• Is a process of having the same name for different methods as long as their parameter list or data types in the parameter list is different?
• The return type alone is insufficient to distinguish two versions of a method.
• Parameters are key identifiers of OM.
• Is one of the ways that Java supports polymorphism

Variant 1 | Alternative version 1

What is method overloading in Java?

A. Creating multiple methods with the same name but different return types.

B. Creating multiple methods with the same name but different parameter lists.

C. Creating multiple methods with the same name and parameter list but different return types.

D. Creating multiple methods with the same name and parameter list but different access modifiers.

Answer B: Creating multiple methods with the same name but different parameter lists.

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Incorrect: Same-name/different-return methods are not necessarily overloads; return type alone is explicitly insufficient

B - Correct: Same name with different parameter lists is the exact overloading definition

C - Incorrect: Same parameter list prevents return-type-only overloading

D - Incorrect: Access-modifier differences do not supply the required parameter difference

Primary note proof

Study-note section OOP12

> Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++.

Study-note section OOP12

> A return type is not part of an ordinary Java method signature.

Related reading (context, not primary note proof)

Study-note section OOP03

Question sources

Computer Science MCQ practice collection · Question 312

Computer Science MCQ practice collection · Question 330

Question 200 | AM0313

What is the purpose of the new keyword in Java?

A. To instantiate a new object.

B. To invoke a method.

C. To define a variable.

D. To define a class.

Hint

Separate a reference declaration, object allocation, reference copying and changes to the shared object.

Answer A: To instantiate a new object.

A reference declaration is distinct from new Dog(), which creates the actual instance and triggers initialization.

Why each choice is right or wrong

A - Correct: Matches object creation

B - Incorrect: An ordinary method call does not by itself state the new allocation

C - Incorrect: A variable declaration can exist without an object

D - Incorrect: class declares the blueprint, not its instance

Primary note proof

Study-note section OOP03

> Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Study-note section OOP09

A constructor has the class’s exact name and no return type, not even void. It initializes the new object and can be overloaded or private.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 8 | lines 43-55

• a class declaration only creates a template; it does not create an actual object.
• To actually create a Dog object, use the second statement.
• Dog myDog; // Creates only the template
• myDog = new Dog(); // create a Dog object called myDog
• After this statement executes, myDog will be an instance of Dog.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Question sources

Computer Science MCQ practice collection · Question 313

Question 201 | AM0314

What is the output of the following Java code snippet?

```
class MyClass {
public static void main(String[] args) {
int x = 5;
System.out.println(x++);
System.out.println(++x);
}
}
```

A. 5, 6

B. 5, 7

C. 6, 6

D. 6, 7

Hint

Track the old and updated values separately and distinguish a character operand from a string.

Answer B: 5, then 7, on separate lines.

OOP24 gives this exact sequence: postfix x++ supplies the old value 5 and then stores 6; prefix ++x then stores and supplies 7. The two println calls print those values on separate lines. The commas in the choices describe the sequence rather than literal commas emitted by the program.

Why each choice is right or wrong

A - Incorrect: The second prefix increment advances the stored 6 to 7 before printing, so the second value is not 6.

B - Correct: Postfix prints the old 5, then prefix increments the stored 6 and prints 7.

C - Incorrect: The first postfix expression supplies 5 rather than the incremented 6.

D - Incorrect: The second value is 7, but the first remains 5 because its increment is postfix.

Primary note proof

Study-note section OOP24

> Starting x=5, printing x++ then ++x gives 5 then 7.

Study-note section OOP-S02

> A print call emits its value without inserting a separator or ending the line. A println call emits its value followed by a line terminator. Spaces must come from the supplied text or explicit output operations; adjacent calls do not automatically separate their values.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 4 | lines 21-34

✓ Operators: Are symbols that take one or more arguments (operands) and operates on them to produce a result. An operator performs a function on one (Unary), two (Binary), or three operands (Ternary). Java has one ternary operator, ?:, which is a short-hand if-else statement. E.g. c=x?w:s; Operators can be Arithmetic, Logical, assignment, Relational, conditional, Increment,

Question sources

Computer Science MCQ practice collection · Question 314

Question 202 | AM0315

Which Java keyword directly exits the enclosing loop prematurely while allowing execution to continue after that loop in the same method?

A. break

B. continue

C. return

D. exit

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer A: break

break exits the enclosing loop and transfers control to the statement after it. continue skips the remaining body of the current iteration and may begin another iteration. return leaves the entire method, so it does not satisfy the stated requirement to continue after the loop in that same method. exit is not a Java loop-control keyword. The narrower control-flow target makes break unique.

Why each choice is right or wrong

A - Correct: break exits the loop and resumes after it within the method.

B - Incorrect: continue proceeds toward another iteration rather than exiting the loop.

C - Incorrect: return exits the entire method, contrary to the required continuation.

D - Incorrect: exit is not a Java loop-control keyword.

Primary note proof

Study-note section OOP25

> break exits the loop; continue skips its remaining body and then runs a for update.

Study-note section OOP-S03

> An unlabeled break leaves the innermost enclosing loop or switch. continue moves to the next iteration step, while return leaves the current method and may supply a return value. These transfers have different destinations; return is not merely a loop exit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 7 | lines 26-41

> Iteration: The for(), while() and
> do...while() statements
> 4. Jumping Statements: to jump out
> of loops and to control other areas of
> program java uses break and
> continue statements.
> The break statement: used to exit a
> loop
> The Continue Statement: used to
> force program to go back to the top
> of a loop
> Labels: Java does not include a goto
> statement.
> Instead of goto, Java allows you to
> combine break and continue with a
> label.

Question sources

Computer Science MCQ practice collection · Question 315

Question 203 | AM0316

What is the purpose of the try-catch block in Java?

A. To handle exceptions and prevent program crashes.

B. To define a custom class.

C. To declare variables with restricted access.

D. To instantiate objects.

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer A: To handle exceptions and prevent program crashes.

An exception disrupts execution, and try/catch monitors code and supplies a handler, as shown for an input mismatch. It can handle matched exceptions, not guarantee prevention of every crash.

Why each choice is right or wrong

A - Correct: Matches exception handling, with no universal recovery guarantee

B - Incorrect: Class definition is a different construct

C - Incorrect: Visibility modifiers govern access

D - Incorrect: new and constructors instantiate objects

Primary note proof

Study-note section OOP30

> try encloses guarded operations; a matching catch handles the thrown type or a subtype. Execution skips the remainder of the try when a throw occurs.

Study-note section OOP03

> Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 27 | lines 20-41

> • An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.
> • An exception is an indication of a problem that occurs during a program‟s execution.
> • An exception is a runtime error.
> • A Java exception is an instance of a class derived from Throwable.

• The Throwable class is contained in the java.lang package, and subclasses of Throwable are contained in various packages.
• Errors related to GUI components are included in the java.awt package; numeric exceptions are included in the java.lang package because they are related to the java.lang.Number class.
• You can create your own exception classes by extending Throwable or a subclass of Throwable.

Object Oriented Programming module | PDF page 28 | lines 8-22

```
try {
int num=
reader.nextInt();
System.out.println("The number
is :" + num);
}
catch
(InputMismatchException e){
System.out.println("There is
type mis match :");
}
finally{
System.out.println("This part is
always done");
}} }
```

Object Oriented Programming module | PDF page 29 | lines 11-22

```
Using try and catch...
General form of try/catch:
Try{
//block of code to monitor for
errors
}
Catch(ExceptionType1 exOb ) {
//handler for ExceptionType1
}
Catch(ExceptionType2 exOb ) {
//handler for ExceptionType2
}
```

Question sources

Computer Science MCQ practice collection · Question 316

Question 204 | AM0317

Which independent Java declarations correctly create an int array? Select all that apply.

A. int[] numbers = {1, 2, 3};

B. int numbers[] = new int[3];

C. int numbers = new int[]{1, 2, 3};

D. int numbers[] = {1, 2, 3};

Hint

Check declared dimensions, initializer shape, fixed length and zero-based index bounds.

Answer A, B, D: int[] numbers = {1, 2, 3};; int numbers[] = new int[3];; int numbers[] = {1, 2, 3};

A and D use legal bracket placements with a brace initializer to create arrays containing 1, 2 and 3. B also declares an int array and allocates three elements, initially zero. C instead declares numbers as a scalar int but supplies an int-array value, producing a type mismatch. Both bracket placement styles and both allocation forms are allowed, so A, B and D are selected.

Why each choice is right or wrong

A - Correct: The int[] type and brace initializer create a three-element int array.

B - Correct: Brackets after the variable name and new int[3] are valid.

C - Incorrect: The scalar int variable cannot hold the int[] initializer value.

D - Correct: The name[] form with a brace initializer is valid.

Primary note proof

Study-note section OOP26

> int[] a=new int[5]; and int a[]={1,2,3}; are valid. int[] a={}; creates a zero-length array; explicit new is unnecessary in a declaration initializer. int a[],b; makes only a an array; int[] c,d; makes both arrays. Array objects have fixed length after creation, although the reference may be reassigned.

Study-note section OOP05

> Java fields and array elements receive defaults: numeric zero, char '\u0000', boolean false and references null.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 6 | lines 4-17

> Java has two basic data types
> 1. Primitive Data Types: are 8 in java
> (byte, short, int, long, float, double,
> char & Boolean)

2. Reference types: Arrays, classes, and interfaces, String)
• Note: Boolean type has a value true or false (there is no conversion b/n Boolean and other types).
• All non-primitive types are reference types, so all class types are reference types.
• All integers in Java are signed, Java doesn‟t support unsigned integers.

Question sources

Computer Science MCQ practice collection · Question 317

Question 205 | AM0318

Which method is automatically called when an object is created in Java?

A. start()

B. run()

C. main()

D. constructor

Hint

Check the class name, return-type absence and the constructors actually declared.

Answer D: constructor

The constructor is automatically called on object creation before new completes.

Why each choice is right or wrong

A - Incorrect: Not the automatic creation routine defined here

B - Incorrect: Not the automatic creation routine defined here

C - Incorrect: main is the program entry method, not every object's initialization

D - Correct: Matches automatic object initialization

Primary note proof

Study-note section OOP09

A constructor has the class’s exact name and no return type, not even void. It initializes the new object and can be overloaded or private.

Study-note section OOP03

Student s=new Student(); allocates and initializes an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Question sources

Computer Science MCQ practice collection · Question 318

Question 206 | AM0319

What is the role of the super keyword in Java?

A. To refer to the current object instance.

B. To invoke a method from the superclass.

C. To define a class as final.

D. To allocate memory for an object.

Hint

Distinguish the current instance, parent context, hidden fields and static dispatch.

Answer B: To invoke a method from the superclass.

super.member may name a method or instance variable of the immediate superclass. This directly covers invoking a superclass method.

Why each choice is right or wrong

A - Incorrect: Current-object role is this rather than super

B - Correct: Matches superclass method access

C - Incorrect: final is the separate class modifier

D - Incorrect: new is the allocation operation

Primary note proof

Study-note section OOP16

> Java this denotes the current object; this.name=name distinguishes a field from a parameter. super.name selects a hidden superclass field; super.sound() directly requests the parent implementation. It does not create a second parent object.

Study-note section OOP07

> final prevents reassignment of a variable, overriding of a method or extension of a class.

Study-note section OOP03

> Student s=new Student(); allocates and initializes an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 17 | lines 51-74

> The super keyword
> • Since encapsulation is a primary attribute of OOP, Whenever a subclass needs to refer to its immediate superclass, it can do so by use of the keyword super.
> • Super has two general forms.
> 1. Used to access parent class instance members.
> 2. Used to access parent class constructor.
> 1. Used to access parent class instance members.
> • super acts somewhat like this, except that it always refers to the superclass of the subclass in which it is used.
> • Super has the form: super.member Here, member can be either a method or an

instance variable.
• Super is most applicable to resolve member hiding.
• super( ) must always be the first statement executed inside a subclass‟ constructor.

Question sources

Computer Science MCQ practice collection · Question 319

Question 207 | AM0320

What does the static keyword mean in Java?

A. It denotes that a variable or method belongs to the superclass.

B. It indicates that a variable or method can only be accessed within the same class.

C. It allows a variable or method to be accessed without creating an instance of the class.

D. It specifies that a variable or method cannot be modified.

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer C: It allows a variable or method to be accessed without creating an instance of the class.

Static members belong to the class, and the static method is accessible by class name without an object. Visibility and immutability are separate properties.

Why each choice is right or wrong

A - Incorrect: Class membership does not mean belonging to the superclass

B - Incorrect: Private is the visibility restriction

C - Correct: Matches class access without creating an instance

D - Incorrect: final, not static, prevents modification

Primary note proof

Study-note section OOP07

static gives one field per loaded class, shared by its instances. If count starts at zero and every constructor increments it, three creations leave count=3. Prefer ClassName.count and ClassName.utility(). A static method has no current instance, but can still use instance members through an explicit object reference.

Study-note section OOP07

final prevents reassignment of a variable, overriding of a method or extension of a class.

Study-note section OOP08

For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 14 | lines 2-19

Static methods and Final Variables
• A static method can access only static data.
• A static method can call only other static data (Static methods & variables), but can not call a non-static data from it.
• Static method is belongs to the class and not to the object.
• A static method can be accessed directly by the class name and doesn‟t need any object.
• i.e ClassName.methodName(args)
• A static method cannot refer to "this" or "super" keywords in anyway
• But can be overloaded since they are resolved using static binding by compiler at compile time.

Object Oriented Programming module | PDF page 5 | lines 39-61

C. Class/Static variables
• Are declared with the static keyword in a class, but outside a method, constructor or a block.
• There would only be one copy of each class variable per class, regardless of how many objects are created from it.
• Are rarely used other than being declared as constants. (public static final)
• Are stored in the static memory.
• Static variables are created when the program starts and destroyed when the program stops.
• Visibility is similar to instance

variables. However, most static variables are declared public
• Default values are same as instance variables.
• Static variables can be accessed by calling with the class name ClassName.VariableName.

Question sources

Computer Science MCQ practice collection · Question 320

Question 208 | AM0321

What is the output of the following Java code snippet?

```
class MyClass {
public static void main(String[] args) {
String text = "Hello";
text += " World";
System.out.println(text);
}
}
```

A. Hello

B. World

C. Hello World

D. Compile error

Hint

Separate identity, content equality, immutable values and references that may be reassigned.

Answer C: Hello World.

OOP22 directly explains that text += " World" assigns a reference to a new concatenated result rather than mutating the original String. The variable therefore refers to "Hello World" before printing. String immutability does not prevent rebinding a non-final variable, so the code is valid and prints the combined text with its space.

Why each choice is right or wrong

A - Incorrect: The variable is rebound to the concatenated result, so it no longer prints only its initial text.

B - Incorrect: Concatenation retains Hello and appends the space and World; it does not replace the entire content with World.

C - Correct: The new concatenated String contains Hello, a space and World, and text refers to that result.

D - Incorrect: String immutability permits this reference reassignment; the code does not attempt to mutate the original String.

Primary note proof

Study-note section OOP22

> Java String objects are immutable. text += " World" assigns a reference to a new concatenated result and can print Hello World without mutating the original String object. final class String prevents subclassing; final alone does not make arbitrary objects immutable.

Study-note section OOP24

> Once a String operand is encountered, + concatenates in left-to-right groups: 1+2+"x" gives "3x", while "x"+1+2 gives "x12".

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 9 | lines 14-16

> 5. String Class:- Immutable
> Class created to manipulate
> strings(Sequence of chars).

Object Oriented Programming module | PDF page 10 | lines 15-26

> • Every Dog object will contain its
> own copies of the instance variables
> breed, age, and color.
> • To access these variables, you will
> use the dot (.) operator.
> • The dot operator links the name of
> the object with the name of an
> instance variable.
> • For example, to assign the age
> variable of myDog the value 10, you
> would use the following statement:
> myDog.age = 10; //This is called

Object Oriented Programming module | PDF page 4 | lines 21-34

> ✓ Operators: Are symbols that
> take one or more arguments
> (operands) and operates on
> them to produce a result. An
> operator performs a function
> on one (Unary), two (Binary),
> or three operands (Ternary).
> Java has one ternary operator,
> ?:, which is a short-hand if-
> else statement. E.g. c=x?w:s;

Operators can be Arithmetic,
Logical, assignment,
Relational, conditional,
Increment,

Question sources

Computer Science MCQ practice collection · Question 321

Question 209 | AM0322

Which access modifier allows a class member to be accessible only within the same package?

A. public

B. private

C. protected

D. Default (no modifier specified)

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer D: Default (no modifier specified)

No modifier/default is explicitly package-private. Other modifiers are narrower or wider than the package-only requirement.

Why each choice is right or wrong

A - Incorrect: World-visible, too broad

B - Incorrect: Class-only, too narrow

C - Incorrect: Also includes out-of-package subclasses

D - Correct: Exactly the package-private scope

Primary note proof

Study-note section OOP08

For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP08

Across packages, a subclass accessing a protected instance member must do so through its own inherited context or an appropriate subclass-typed receiver, not any arbitrary superclass object. protected therefore does not mean “subclasses only”: unrelated classes in the same package also have access.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

Access Specifiers
Access level modifiers determine whether other classes can use a particular field or invoke a particular method.
Their are four types access levels:
1. No Modifier(Default): Visible to the package(source file), and it is called package-private
2. Private: Visible to the class only.
3. Public: Visible to the world.
4. Protected: Visible to the package and all subclasses.

Question sources

Computer Science MCQ practice collection · Question 322

Question 210 | AM0323

What is the output of the following Java code snippet?

```
class MyClass {
private int x = 5;

public int getX() {
return x;
}

public static void main(String[] args) {
MyClass obj = new MyClass();
obj.x = 10;
System.out.println(obj.getX());
}
}
```

A. 5

B. 10

C. Compile error

D. Runtime exception

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer B: 10

The write to obj.x is inside MyClass itself, even though main is static; private is class-based, not object-based. obj.x=10 updates the instance, and getX returns that field.

Why each choice is right or wrong

A - Incorrect: Old initializer is replaced by the assignment

B - Correct: Getter returns the updated field value 10

C - Incorrect: Same-class code may access private fields through obj

D - Incorrect: No exceptional operation is indicated by these class-local accesses

Primary note proof

Study-note section OOP08

> A class's own main method can access private fields of an instance of that class.

Study-note section OOP07

> A static method has no current instance, but can still use instance members through an explicit object reference.

Study-note section OOP11

> An accessor/getter returns information; a mutator/setter changes state.

Study-note section OOP-S03

> A later assignment overwrites an earlier value.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether
> other classes can use a particular field or
> invoke a particular method.
> Their are four types access levels:
> 1. No Modifier(Default): Visible to
> the package(source file), and it is
> called package-private
> 2. Private: Visible to the class only.
> 3. Public: Visible to the world.
> 4. Protected: Visible to the package
> and all subclasses.

Object Oriented Programming module | PDF page 10 | lines 15-26

> • Every Dog object will contain its
> own copies of the instance variables

breed, age, and color.
• To access these variables, you will use the dot (.) operator.
• The dot operator links the name of the object with the name of an instance variable.
• For example, to assign the age variable of myDog the value 10, you would use the following statement: myDog.age = 10; //This is called

Question sources

Computer Science MCQ practice collection · Question 323

Question 211 | AM0324

What is the purpose of the toString() method in Java?

A. To convert an object to a string representation.

B. To compare two objects for equality.

C. To perform arithmetic calculations.

D. To access the superclass methods.

Hint

Separate identity, content equality, immutable values and references that may be reassigned.

Answer A: Return an object’s textual representation.

OOP22 defines toString as giving a textual representation and distinguishes it from an equality test. A states that purpose. Comparing objects is the role of equals under the class’s policy, while arithmetic and superclass method access are separate operations. The representation need not create a new copy of the original object.

Why each choice is right or wrong

A - Correct: A textual String representation of an object is the purpose explicitly assigned to toString in OOP22.

B - Incorrect: Object equality is handled by equals under the class’s equality policy, not by toString.

C - Incorrect: Arithmetic calculations are not the defining contract of an object’s textual representation method.

D - Incorrect: Superclass access uses super; it is not the purpose of toString.

Primary note proof

Study-note section OOP22

For object references, == tests identity; equals invokes the class's equality policy. Object.equals uses identity unless overridden; String.equals compares character content. Equal objects must have equal hashCode values when that contract is implemented. toString gives a textual representation, not an equality test.

Study-note section OOP16

super.name selects a hidden superclass field; super.sound() directly requests the parent implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 10 | lines 62-72

Methods
• Are members of classes and do
operations.
• Allows different classes to
communicate each other.
• E.g. type name(parameter-
list)
{ // body of method
//....
return value; // if
type is not void

Object Oriented Programming module | PDF page 9 | lines 14-16

5. String Class:- Immutable
Class created to manipulate
strings(Sequence of chars).

Question sources

Computer Science MCQ practice collection · Question 324

Question 212 | AM0325

What is the purpose of the equals() method in Java?

A. To convert an object to a string representation.

B. To compare two objects for equality.

C. To perform arithmetic calculations.

D. To access the superclass methods.

Hint

Separate identity, content equality, immutable values and references that may be reassigned.

Answer B: Compare objects according to the class's equality policy.

OOP22 says equals invokes a class's equality policy; Object.equals uses identity unless overridden, while String.equals compares content. B is therefore correct without assuming every class implements value equality. The other choices describe textual representation, arithmetic or superclass access, none of which is equals' comparison purpose.

Why each choice is right or wrong

A - Incorrect: Textual representation is the purpose of toString, rather than equals.

B - Correct: equals compares objects using the class's policy, which may be identity or an overridden value/content policy.

C - Incorrect: equals supplies an equality result rather than being an arithmetic calculation mechanism.

D - Incorrect: Superclass access uses super and is distinct from the equality comparison performed by equals.

Primary note proof

Study-note section OOP22

> For object references, == tests identity; equals invokes the class's equality policy. Object.equals uses identity unless overridden; String.equals compares character content. Equal objects must have equal hashCode values when that contract is implemented. toString gives a textual representation, not an equality test.

Study-note section OOP16

> super.name selects a hidden superclass field; super.sound() directly requests the parent implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 10 | lines 62-72

> Methods
> • Are members of classes and do
> operations.
> • Allows different classes to
> communicate each other.
> • E.g. type name(parameter-
> list)
> { // body of method
> //....
> return value; // if
> type is not void

Question sources

Computer Science MCQ practice collection · Question 325

Question 213 | AM0326

What is the correct way to create an instance of a class in Java?

A. MyClass obj = new MyClass();

B. MyClass obj = MyClass();

C. MyClass obj = create MyClass();

D. MyClass obj = instance MyClass();

Hint

Separate a reference declaration, object allocation, reference copying and changes to the shared object.

Answer A: MyClass obj = new MyClass();

Dog myDog = new Dog(); includes a declaration, new allocation, constructor parentheses and assignment. A follows precisely that form.

Why each choice is right or wrong

A - Correct: Matches the demonstrated object construction

B - Incorrect: Omission of new does not match construction syntax

C - Incorrect: create is not Java's allocation keyword

D - Incorrect: instance is not Java's allocation keyword

Primary note proof

Study-note section OOP03

> Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Study-note section OOP09

> A constructor has the class's exact name and no return type, not even void. It initializes the new object and can be overloaded or private.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 8 | lines 43-55

> • a class declaration only creates a template; it does not create an actual object.
> • To actually create a Dog object, use the second statement.
> • Dog myDog; // Creates only the template

• myDog = new Dog(); //
create a Dog object called
myDog
• After this statement executes,
myDog will be an instance of
Dog.

Question sources

Computer Science MCQ practice collection · Question 326

Question 214 | AM0327

What is the purpose of the static keyword for a method in Java?

A. It indicates that the method can be overridden.

B. It allows the method to be accessed without creating an instance of the class.

C. It denotes that the method can only be called by the class itself.

D. It specifies that the method cannot be modified.

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer B: It allows the method to be accessed without creating an instance of the class.

Static methods can be called directly by class name without an object and cannot be overridden; a subclass declaration can hide them.

Why each choice is right or wrong

A - Incorrect: Contradicts the static non-overriding rule

B - Correct: Matches object-free class access

C - Incorrect: Visibility depends on access modifiers, not static alone

D - Incorrect: Immutability/non-overriding is not the same as prohibition on modifying the method

Primary note proof

Study-note section OOP07

static gives one field per loaded class, shared by its instances. If count starts at zero and every constructor increments it, three creations leave count=3. Prefer ClassName.count and ClassName.utility(). A static method has no current instance, but can still use instance members through an explicit object reference.

Study-note section OOP14

Constructors, final methods, static methods and private parent methods are not overridden.

Study-note section OOP08

For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 14 | lines 2-19

Static methods and Final Variables
• A static method can access only static data.
• A static method can call only other static data (Static methods & variables), but can not call a non-static data from it.
• Static method is belongs to the class and not to the object.
• A static method can be accessed directly by the class name and doesn‟t need any object.
• i.e ClassName.methodName(args)
• A static method cannot refer to "this" or "super" keywords in anyway
• But can be overloaded since they are resolved using static binding by compiler at compile time.

Object Oriented Programming module | PDF page 21 | lines 26-39

Rules of method overriding in Java..
• Argument list: The argument list of overriding (method of subclass) and Overridden (method of super class) method must be similar.
• The data types of the arguments and their sequence should exactly match.
• Method names: must be similar.
• private, static and final methods cannot be overridden as they are local to the class.
• Access Modifier of the overriding method cannot be more restrictive than the overridden method of parent class.

Question sources

Computer Science MCQ practice collection · Question 327

Question 215 | AM0328

What is the role of the this keyword in Java?

A. It refers to the current object instance.

B. It refers to the superclass of the current class.

C. It refers to the class that contains the current method.

D. It refers to the class from which the current class is derived.

Hint

Distinguish the current instance, parent context, hidden fields and static dispatch.

Answer A: this refers to the current object instance.

OOP16 defines this as the current object. In the main options that statement is A; it denotes an instance rather than its class, subclass or superclass. The source variant moves the same correct statement to C, so its labels must be reviewed separately. super is the distinct parent-context mechanism.

Why each choice is right or wrong

A - Correct: The current object instance is exactly what Java this denotes in ordinary instance context.

B - Incorrect: Superclass context is selected by super, not by the current-object reference this.

C - Incorrect: this refers to an instance, not to the class declaration containing a method.

D - Incorrect: The class from which the current class derives is its superclass, not the meaning of this.

Primary note proof

Study-note section OOP16

> Java this denotes the current object; this.name=name distinguishes a field from a parameter. super.name selects a hidden superclass field; super.sound() directly requests the parent implementation. It does not create a second parent object.

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 17 | lines 51-74

The super keyword
• Since encapsulation is a primary attribute of OOP, Whenever a subclass needs to refer to its immediate superclass, it can do so by use of the keyword super.
• Super has two general forms.
1. Used to access parent class instance members.
2. Used to access parent class constructor.
1. Used to access parent class instance members.
• super acts somewhat like this, except that it always refers to the superclass of the subclass in which it is used.
• Super has the form: super.member Here, member can be either a method or an instance variable.
• Super is most applicable to resolve member hiding.
• super( ) must always be the first statement executed inside a subclass‟ constructor.

Variant 1 | Alternative version 1

What is the role of the this keyword in Java?

A. It refers to the superclass of the current class.

B. It refers to the subclass of the current class.

C. It refers to the current object instance.

D. It refers to the class that contains the current method.

Answer C: this refers to the current object instance.

OOP16 defines this as the current object. This source variant places that definition at C, rather than main-option A. The superclass and subclass alternatives name type relationships, and the containing-class alternative denotes a type rather than the current instance.

Why each choice is right or wrong

A - Incorrect: Superclass context is associated with super, rather than the current-instance reference this.

B - Incorrect: A subclass is a type relationship; this denotes the current object, not a subclass.

C - Correct: This variant places the exact current-object-instance definition from OOP16 at label C.

D - Incorrect: The containing class is a type; this refers to the current object instance of a class.

Primary note proof

Study-note section OOP16

> Java this denotes the current object; this.name=name distinguishes a field from a parameter. super.name selects a hidden superclass field; super.sound() directly requests the parent implementation. It does not create a second parent object.

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent.

Question sources

Computer Science MCQ practice collection · Question 328

Computer Science MCQ practice collection · Question 342

Question 216 | AM0329

What is a constructor in Java?

A. A method that returns a value.

B. A method used for exception handling.

C. A special method used to initialize objects.

D. A method used to override a superclass method.

Hint

Check the class name, return-type absence and the constructors actually declared.

Answer C: A special method used to initialize objects.

Constructors automatically initialize objects, have the same name as their class and have no return type.

Why each choice is right or wrong

A - Incorrect: Contradicts the no-return-type rule

B - Incorrect: Exception handling uses try/catch

C - Correct: Matches automatic initialization of new objects

D - Incorrect: Constructors are not inherited overriding methods

Primary note proof

Study-note section OOP09

> A constructor has the class's exact name and no return type, not even void. It initializes the new object and can be overloaded or private.

Study-note section OOP-S05

> In Java's class-declaration terminology, constructors are not members and are not inherited.

Study-note section OOP30

try encloses guarded operations; a matching catch handles the thrown type or a subtype. Execution skips the remainder of the try when a throw occurs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Question sources

Computer Science MCQ practice collection · Question 329

Question 217 | AM0331

Assume an accessible inherited instance method is non-final, the subclass supplies the same name and parameter list with a compatible return type, and visibility is not reduced. Which descriptions fit the resulting method overriding? Select all that apply.

A. Creating multiple methods with the same name but different return types.

B. Creating multiple methods with the same name but different parameter lists.

C. Creating a new method with the same name and parameter list in the subclass.

D. Redefining a method in the subclass that already exists in the superclass.

Hint

Check inheritance, signature compatibility, visibility and eligibility for an override.

Answer C, D: Creating a new method with the same name and parameter list in the subclass.; Redefining a method in the subclass that already exists in the superclass.

Under the stated eligibility, signature, return and visibility conditions, the subclass method overrides the inherited method. C describes the same name and parameters, and D describes redefining the inherited superclass operation, so both describe this case. Different parameter lists denote overloading instead. Merely changing return types does not define overriding or a valid overload. The assumptions make the two overlapping valid descriptions explicit.

Why each choice is right or wrong

A - Incorrect: Different return types alone do not define overriding or distinguish valid overloads.

B - Incorrect: Different parameter lists describe overloading rather than this same-signature override.

C - Correct: The same-name, same-parameter subclass declaration is an override under the supplied conditions.

D - Correct: It redefines the eligible inherited superclass operation under the supplied conditions.

Primary note proof

Study-note section OOP14

> A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype. Visibility cannot become more restrictive, and checked exceptions cannot be broadened incompatibly.

Study-note section OOP12

> Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 55-67

> Method Overriding
> In a class hierarchy, when a method in a
> subclass has the same name and type
> signature as a method in its superclass, this
> is called method Overriding.
> • the method in the subclass is said to
> override the method in the
> superclass.
> • The Method in the super class is

known as an overridden method
which is called within a subclass.
• The version of the method defined
by the superclass will be hidden.

Object Oriented Programming module | PDF page 21 | lines 26-39

Rules of method overriding in Java..
• Argument list: The argument list of
overriding (method of subclass) and
Overridden (method of super class)
method must be similar.
• The data types of the arguments and
their sequence should exactly match.
• Method names: must be similar.
• private, static and final methods cannot
be overridden as they are local to the
class.
• Access Modifier of the overriding
method cannot be more restrictive than
the overridden method of parent class.

Question sources

Computer Science MCQ practice collection · Question 331

Question 218 | AM0333

What is polymorphism in Java?

A. The ability to create multiple objects of the same class.

B. The process of converting an object to a string representation.

C. The ability of an object to take on many forms through inheritance.

D. The process of comparing two objects for equality.

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer C: The ability of an object to take on many forms through inheritance.

Polymorphism means many forms, typically illustrated by a superclass reference pointing at child objects; inheritance supplies those related forms.

Why each choice is right or wrong

A - Incorrect: Several objects of the same class is multiplicity of instances

B - Incorrect: Object-to-string conversion is unrelated

C - Correct: Matches many forms through a hierarchy

D - Incorrect: Equality comparison is unrelated

Primary note proof

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile.

Study-note section OOP22

> For object references, == tests identity; equals invokes the class's equality policy. Object.equals uses identity unless overridden; String.equals compares character content. Equal objects must have equal hashCode values when that contract is implemented. toString gives a textual representation, not an equality test.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 25 | lines 17-37

> • Polymorphism is the ability of an object to take on many forms.
> • The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.
> • Any Java object that can pass more than one IS-A test is considered to be polymorphic.
> • In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class Object.
> • Polymorphism is derived from 2 Greek words: poly and morphs.
> • The word "poly" means many and "morphs" means forms. So polymorphism means many forms.
> • polymorphism is the capability of an action or method to do different things based on the object that it is acting upon.

Question sources

Computer Science MCQ practice collection · Question 333

Question 219 | AM0334

Which object-oriented principle promotes the concept of reusing code and creating relationships between classes?

A. Encapsulation

B. Inheritance

C. Polymorphism

D. Abstraction

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer B: Inheritance

The inheritance section explicitly links reuse to creating a new class that inherits Person without changing Person, forming the parent/child relationship.

Why each choice is right or wrong

A - Incorrect: Wraps and hides data rather than deriving related classes

B - Correct: Directly supplies reuse through inherited relationships

C - Incorrect: Uses varying behavior through types rather than creating the reuse relationship

D - Incorrect: Suppresses details rather than acquiring reusable members

Primary note proof

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects. Prefer a precise mechanism over a vague slogan: wrapping fields and methods points to encapsulation; showing what an operation does while hiding how points to abstraction.

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Study-note section OOP13

> Use inheritance when a subtype can satisfy the parent’s contract: Dog is an Animal. For shared Pet.name and speak(), put common state/contract in Pet, specialized fetch() in Dog, and appropriate speak() implementations in Dog and Cat. Composition represents has-a, such as Car containing Engine.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 46-56

2. Inheritance:
• It is the way in which objects of one class get the properties of objects of another class(data &methods).
• For example, Person, employee and retiree
• Inheritance provides the idea of reusability - the Person class can be reused without making changes to it, because another class can be made to inherit from it.

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{

```
.....}
class Sub extends Super{ }
```

Question sources

Computer Science MCQ practice collection · Question 334

Question 220 | AM0335

What is the purpose of encapsulation in object-oriented programming?

A. To create relationships between classes

B. To allow classes to inherit properties and behaviors

C. To hide internal implementation details and expose a public interface

D. To provide different implementations of a method based on the object's type

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer C: To hide internal implementation details and expose a public interface

The hidden-data part of encapsulation exposes access through methods, which is the public interaction boundary described by C. A/B describe inheritance and D describes polymorphic overriding.

Why each choice is right or wrong

A - Incorrect: Describes class relationships generally

B - Incorrect: Describes inheritance specifically

C - Correct: Matches hidden data accessed through methods

D - Incorrect: Describes different behavior based on receiver type

Primary note proof

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller. These ideas overlap: a private balance plus deposit(amount) can demonstrate all three, depending on what the question emphasizes.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 35-45

ꢁ. Encapsulation:
• means the wrapping up of data and methods into a single unit (a class).
• It also means that the data inside a class is hidden from everything outside the class.
• The data can only be accessed by invoking the methods of the class.
• It is placing the data and the functions that work on that data in the same place.

Object Oriented Programming module | PDF page 3 | lines 27-33

ꢀ. Abstraction:
• provides only essential information to the outside world.
• Or hiding their background details, i.e., to represent the needed information in program without presenting the details.

Question sources

Computer Science MCQ practice collection · Question 335

Question 221 | AM0336

Which object-oriented principle allows an object to exhibit different behaviors based on its data type?

A. Encapsulation

B. Inheritance

C. Polymorphism

D. Abstraction

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer C: Polymorphism

An operation can behave differently for different subclasses, with method behavior determined by the object acted upon.

Why each choice is right or wrong

A - Incorrect: Groups and hides data

B - Incorrect: Creates the subtype relation but does not name varying operation behavior

C - Correct: Matches behavior varying with object type

D - Incorrect: Shows essentials while suppressing details

Primary note proof

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Study-note section OOP15

> For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 58-64

> 2. Polymorphism:
> • The word polymorphism means the ability to take more than one form.
> • In terms of the OOP, this means that a particular operation may behave differently for different sub-classes of the same class.

Object Oriented Programming module | PDF page 25 | lines 17-37

> • Polymorphism is the ability of an object to take on many forms.
> • The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.
> • Any Java object that can pass more than one IS-A test is considered to be polymorphic.
> • In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class

Object.
• Polymorphism is derived from 2 Greek words: poly and morphs.
• The word "poly" means many and "morphs" means forms. So polymorphism means many forms.
• polymorphism is the capability of an action or method to do different things based on the object that it is acting upon.

Question sources

Computer Science MCQ practice collection · Question 336

Question 222 | AM0337

What is the purpose of abstraction in object-oriented programming?

A. To create relationships between classes

B. To allow classes to inherit properties and behaviors

C. To hide internal implementation details and expose a public interface

D. To provide different implementations of a method based on the object's type

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer C: To hide internal implementation details and expose a public interface

Abstraction represents needed/essential information while hiding background details, matching a public-facing interface to hidden implementation.

Why each choice is right or wrong

A - Incorrect: Describes relationships rather than the essential view

B - Incorrect: Describes inheritance

C - Correct: Matches essential information with background details hidden

D - Incorrect: Describes polymorphic differing implementations

Primary note proof

Study-note section OOP02

Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller. These ideas overlap: a private balance plus deposit(amount) can demonstrate all three, depending on what the question emphasizes.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 27-33

> 0. Abstraction:
> • provides only essential information
> to the outside world.
> • Or hiding their background details,
> i.e., to represent the needed
> information in program without
> presenting the details.

Question sources

Computer Science MCQ practice collection · Question 337

Question 223 | AM0338

Which Java keyword declares that one class inherits from another class?

A. super

B. extends

C. implements

D. this

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer B: extends

A class declaration uses extends to name its direct superclass, as in class Dog extends Animal. A class implements interface contracts using implements, which is a different declaration relationship. super selects parent context inside a class, while this denotes the current instance. Restricting the question to class-to-class inheritance makes extends the unique keyword among these choices.

Why each choice is right or wrong

A - Incorrect: Refers to superclass members/constructor

B - Correct: Matches class Sub extends Super

C - Incorrect: Used for interface implementation instead

D - Incorrect: Refers to current-instance context, not an inheritance declaration

Primary note proof

Study-note section OOP13

class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent. A class extends one direct class and can implement multiple interfaces.

Study-note section OOP16

Java this denotes the current object; this.name=name distinguishes a field from a parameter. super.name selects a hidden superclass field; super.sound() directly requests the parent implementation.

Study-note section OOP19

A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends

keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Object Oriented Programming module | PDF page 31 | lines 9-30

• It provides total abstraction;
• Means all the methods in interface are declared with empty body and are public and all fields are public, static and final by default.
• A class that implement interface must implement all the methods declared in the interface.
• If a class implements an interface and does not provide method bodies for all functions specified in the interface, then class must be declared abstract.
Rules of Interface
• Variables declared in an interface are public, static & final by default.
• Java does not allow you to extend more than one class, However you can implement more than one interfaces in your class.
• class implements interface but an interface extends another interfaces.

Question sources

Computer Science MCQ practice collection · Question 338

Question 224 | AM0339

Which Java annotation requests compiler checking that a method overrides or implements an inherited method contract?

A. super

B. extends

C. implements

D. @Override

Hint

Check inheritance, signature compatibility, visibility and eligibility for an override.

Answer D: @Override

@Override asks the compiler to verify that a declaration satisfies overriding rules. Java does not have an override keyword, and the annotation does not create an override by itself; the declaration must still meet the inherited-contract conditions. super selects parent context, extends declares class inheritance, and implements declares interface relationships. Correcting the requested construct to an annotation preserves the intended override-checking concept.

Why each choice is right or wrong

A - Incorrect: super selects parent members or construction rather than checking a declaration's override status.

B - Incorrect: extends establishes class inheritance rather than annotating a method.

C - Incorrect: implements establishes an interface relationship rather than checking one method declaration.

D - Correct: @Override requests compiler verification of the overriding/implementation relationship.

Primary note proof

Study-note section OOP14

> A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype. Visibility cannot become more restrictive, and checked exceptions cannot be broadened incompatibly. @Override requests compiler checking; Java has no override keyword. Constructors, final methods, static methods and private parent methods are not overridden.

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent. A class extends one direct class and can implement multiple interfaces.

Study-note section OOP16

> super.name selects a hidden superclass field; super.sound() directly requests the parent implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 55-67

> Method Overriding
> In a class hierarchy, when a method in a
> subclass has the same name and type
> signature as a method in its superclass, this
> is called method Overriding.
> • the method in the subclass is said to
> override the method in the
> superclass.

• The Method in the super class is known as an overridden method which is called within a subclass.
• The version of the method defined by the superclass will be hidden.

Object Oriented Programming module | PDF page 15 | lines 23-53

What is Inheritance?
• Inheritance is one of the cornerstones of object-oriented programming because it allows the creation of hierarchical classifications.
• Inheritance is the process where one class acquires the properties (methods and fields) of another.
• A class that is inherited is called a superclass.
• The class that does the inheriting is called a subclass.
• OR: The class which inherits the properties of other is known as subclass (derived class, child class) and
• The class whose properties are inherited is known as superclass (base class, parent class).
• Therefore, a subclass is a specialized version of a superclass.
• It inherits all of the instance variables and methods defined by the superclass and add its own, unique elements.
• The extends keyword: is the keyword used to inherit the properties of a class. Below given is the syntax of extends keyword.
• Sytax:
class Super{
.....}
class Sub extends Super{ }

Object Oriented Programming module | PDF page 21 | lines 26-39

Rules of method overriding in Java..
• Argument list: The argument list of overriding (method of subclass) and Overridden (method of super class) method must be similar.
• The data types of the arguments and

their sequence should exactly match.
• Method names: must be similar.
• private, static and final methods cannot be overridden as they are local to the class.
• Access Modifier of the overriding method cannot be more restrictive than the overridden method of parent class.

Question sources

Computer Science MCQ practice collection · Question 339

Question 225 | AM0341

Which object-oriented principle promotes the concept of grouping related properties and behaviors into a single unit?

A. Encapsulation

B. Inheritance

C. Polymorphism

D. Abstraction

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer A: Encapsulation

The stem repeats the definition of grouping data and methods into a single class unit.

Why each choice is right or wrong

A - Correct: Direct wrapping-up/grouping principle

B - Incorrect: Acquires another class's members

C - Incorrect: Many behavior forms

D - Incorrect: Essential-information view rather than grouping

Primary note proof

Study-note section OOP02

Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller. These ideas overlap: a private balance plus deposit(amount) can demonstrate all three, depending on what the question emphasizes.

Study-note section OOP02

Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 3 | lines 35-45

ვ. Encapsulation:
• means the wrapping up of data and methods into a single unit (a class).
• It also means that the data inside a class is hidden from everything outside the class.
• The data can only be accessed by invoking the methods of the class.
• It is placing the data and the functions that work on that data in the same place.

Question sources

Computer Science MCQ practice collection · Question 341

Question 226 | AM0343

Which object-oriented concept is used to create objects based on a predefined blueprint or class?

A. Polymorphism

B. Inheritance

C. Encapsulation

D. Instantiation

Hint

Separate a reference declaration, object allocation, reference copying and changes to the shared object.

Answer D: Instantiation

A class is the template and an object its instance; new creates that actual object. Instantiation names creating the instance from the template.

Why each choice is right or wrong

A - Incorrect: Many forms rather than creation from a blueprint

B - Incorrect: Deriving classes rather than objects

C - Incorrect: Grouping data/methods rather than object creation

D - Correct: Matches creating the actual instance from a class template

Primary note proof

Study-note section OOP03

> Student s; declares a reference variable; it creates no Student object. Student s=new Student(); allocates and initializes an instance.

Study-note section OOP02

> Encapsulation groups related state and operations and controls access.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 8 | lines 43-55

> • a class declaration only creates a template; it does not create an actual object.
> • To actually create a Dog object, use the second statement.
> • Dog myDog; // Creates only the template
> • myDog = new Dog(); // create a Dog object called myDog
> • After this statement executes, myDog will be an instance of Dog.

Object Oriented Programming module | PDF page 8 | lines 4-20

> 2.1. Fundamentals of Classes
> • A class is a blue print from which individual objects are created.
> • Class is a template for an object, and an object is an instance of a class
> • Class is the logical construct upon which the entire Java language is built because it defines the shape and nature of an object.
> • As such, the class forms the basis for object-oriented programming in Java.
> • Classes usually consist of two things:

instance variables and methods
• Any concept you wish to implement
in a Java program must be
encapsulated within a class.

Question sources

Computer Science MCQ practice collection · Question 343

Question 227 | CS0100

Which one of the following is true regarding constructors in Java?

A. Constructors are invoked to create objects without using the new operator

B. Constructors can have a return type

C. A constructor can have a different name from its class name

D. Constructors play the role of initializing objects

Hint

Check the class name, return-type absence and the constructors actually declared.

Answer D: Constructors play the role of initializing objects

Constructors initialize objects automatically during new, share the class name and have no return type.

Why each choice is right or wrong

A - Incorrect: Contradicts the described new/constructor creation sequence for ordinary construction

B - Incorrect: Contradicts no return type, not even void

C - Incorrect: Contradicts same-class-name rule

D - Correct: Matches automatic initialization

Primary note proof

Study-note section OOP09

A constructor has the class's exact name and no return type, not even void. It initializes the new object and can be overloaded or private.

Study-note section OOP03

Student s=new Student(); allocates and initializes an instance.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 12 | lines 24-47

Constructors
• Performs automatic initialization of objects. .:.
• Initializes an object immediately upon creation.
• Once defined, the constructor is automatically called immediately after the object is created, before the new operator completes.
• The default constructor automatically initializes all instance variables to zero and all object references to NULL.
• The default constructor is often sufficient for simple classes, but it usually won‟t do for more sophisticated ones.
• Once you define your own constructor, the default constructor is no longer used.
• Syntax:
• Have the same name with their class name.
• Have not return type, not even void.

Question sources

Computer Science Exit Exam practice collection · Question 100

Question 228 | CS0101

If you want to make methods and data fields of classes accessible from any class in any package, which access modifier is appropriate?

A. protected

B. default

C. public

D. private

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer C: public

Public means visible to the world, including unrelated classes in other packages.

Why each choice is right or wrong

A - Incorrect: Only package and subclasses, not every class

B - Incorrect: Only the package

C - Correct: World-visible, satisfying all classes/packages

D - Incorrect: Only the declaring class

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 13 | lines 62-73

> Access Specifiers
> Access level modifiers determine whether
> other classes can use a particular field or
> invoke a particular method.
> Their are four types access levels:
> 1. No Modifier(Default): Visible to
> the package(source file), and it is
> called package-private
> 2. Private: Visible to the class only.
> 3. Public: Visible to the world.
> 4. Protected: Visible to the package
> and all subclasses.

Question sources

Computer Science Exit Exam practice collection · Question 101

Question 229 | CS0102

Which one of the following is NOT correct about function overloading in C++?

A. Two or more overloaded functions may differ in their number of parameters

B. Two or more overloaded functions may differ in data types of their parameters

C. Two or more overloaded functions should have the same function name

D. Two or more overloaded functions may differ only in their return types

Hint

Compare names and parameter signatures before considering return types or access modifiers.

Answer D: Return type alone cannot distinguish C++ overloads.

OOP12 states that overloaded methods/functions share a name and differ in parameter types, counts or order. It expressly rejects return type alone as sufficient in Java and C++. D is therefore the incorrect overloading claim. A, B and C describe valid differences or the shared-name requirement rather than violating the overload rule.

Why each choice is right or wrong

A - Incorrect: Different parameter counts can distinguish overloads, so this is a correct overloading statement.

B - Incorrect: Different parameter data types can distinguish overloads, so this is a correct overloading statement.

C - Incorrect: Overloads share a function name, so this statement correctly describes their naming requirement.

D - Correct: This is the false claim: return type alone cannot distinguish C++ overloads with otherwise identical signatures.

Primary note proof

Study-note section OOP12

> Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Object Oriented Programming module | PDF page 11 | lines 7-18

> Method Overloading
> • Is a process of having the same name for different methods as long as their parameter list or data types in the parameter list is different?
> • The return type alone is insufficient to distinguish two versions of a method.
> • Parameters are key identifiers of OM.
> • Is one of the ways that Java supports polymorphism

Question sources

Computer Science Exit Exam practice collection · Question 102

Question 230 | NS0219

Which purposes apply to Java interfaces? “Blueprint” means an operation contract, “state and behavior” in B means complete per-instance state and implementation, and “multiple inheritance” means type inheritance. Select all that apply.

A. To create multiple constructors for a class

B. To define the state and behavior of an object

C. To define a blueprint of a class

D. To achieve multiple inheritance

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer C, D: To define a blueprint of a class; To achieve multiple inheritance

An interface gives implementing classes an operation contract, so the defined blueprint purpose is valid. A class can implement multiple interfaces, supporting multiple type inheritance. Interfaces have no constructors or per-instance fields, so they do not create a class’s constructors or define an object’s complete state and implementation. With those meanings explicit, C and D are valid purposes and the other two are excluded.

Why each choice is right or wrong

A - Incorrect: Interfaces have no constructors and do not create class constructors.

B - Incorrect: The complete per-instance state and implementation belong to classes, not interface capability contracts.

C - Correct: The blueprint is an operation contract for implementing classes.

D - Correct: Multiple implemented interfaces provide multiple type inheritance.

Primary note proof

Study-note section OOP19

> interface Payable{double getPaymentAmount();} declares an abstract public operation. class Invoice implements Payable must provide public double getPaymentAmount(), perhaps returning getQuantity()*getPricePerItem(). A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces. No interface instance fields or constructors exist; fields are implicitly public static final.

Study-note section OOP20

> Use an abstract class for shared state, construction and common implementation; use an interface for a capability across unrelated classes and multiple type inheritance.

Question sources

Computer Science Exit Exam practice collection · Question 104

Question 231 | NS0292

Consider this C++17 class and statements inside a valid main function:
struct Meter {
int reading = 0;
void set(int reading) { this->reading = reading; }
static int twice(int x) { return 2 * x; }
};
Meter first, second;
first.set(4);
second.set(9);
int doubled = Meter::twice(3);
Which statement correctly describes the implicit object context during second.set(9) and the call to twice?

A. In set, this points to second; twice has no this pointer.

B. In set, this points to first; twice has no this pointer.

C. In set, this points to second; twice reuses second as its implicit object.

D. The hidden object argument is passed to the reading data member rather than to set.

Hint

Use the stated language rules and identify whether a member call has an implicit current object.

Answer A: In set, this points to second; twice has no this pointer.

Each non-static set call has an implicit object: first for first.set(4) and second for second.set(9). During the latter call, this therefore points to second, and this->reading selects that object's field. The static twice function has no current object or this pointer, regardless of which instance method ran most recently. Data members themselves do not receive hidden call arguments.

Why each choice is right or wrong

A - Correct: second is the receiver of second.set(9); the static twice call has no implicit object.

B - Incorrect: The earlier call on first does not make first the receiver of the later call on second.

C - Incorrect: twice is static and does not acquire a this pointer from the most recent instance call.

D - Incorrect: The implicit object belongs to the non-static function call; the reading data member does not receive a hidden argument.

Primary note proof

Study-note section OOP17

> In ordinary C++17 non-static member functions, this is a pointer to the object on which the function is called; this->field selects a member. It is often explained as an implicit object argument to a member function call. It is not an argument passed to member variables, and a static member function has no this pointer.

Question sources

Computer Science Exit Exam · 2018 · Question 9

Question 232 | NS0300

What will be the output of the following program segment?
public class Calculation {

public static void main(String[] args) {
int sum = 0;
for(int j = 1; j<=10; j++) {
sum = sum + j;
}
System.out.println(+sum);
}

A. 55

B. 44

C. 33

D. 66

Hint

Trace condition tests, control transfers and printing as separate steps.

Answer A: 55 for the shown program segment.

The shown program segment initializes sum to zero and adds j for all values 1 through 10 inclusive. Its numerical result is 55. The unary plus in println(+sum) preserves that value. This is a program segment; a complete compilation unit also requires the outer class's closing brace.

Why each choice is right or wrong

A - Correct: The ten inclusive additions produce 55; unary plus leaves that integer unchanged.

B - Incorrect: 44 is not the sum of the stated inclusive sequence 1 through 10.

C - Incorrect: 33 does not match all ten additions performed by the shown loop.

D - Incorrect: 66 would correspond to including an additional 11, which this condition excludes.

Primary note proof

Study-note section OOP25

> Count executions independently from printing.

Study-note section OOP25

> A loop summing j=1..10 yields 55.

Study-note section OOP-S03

Prefix ++ and -- update a variable before supplying its value; postfix forms supply the old value before the update. Multiplication, division and remainder bind before addition and subtraction; same-level binary arithmetic groups left to right. Integer division truncates toward zero. Unary plus preserves its numeric operand’s value.

Question sources

Computer Science Exit Exam · 2018 · Question 17

Question 233 | NS0318

Which OOP principle specifically defines the boundary that groups an object's state with its methods and controls clients' access to that state?

A. Abstraction

B. Inheritance

C. Polymorphism

D. Encapsulation

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer D: Encapsulation.

Encapsulation groups related state and operations and controls access to them. That grouping and access boundary is the mechanism requested. Abstraction emphasizes the essential public view while omitting irrelevant details; it can coexist with encapsulation but does not name this specific boundary. Inheritance forms subtype relationships, and polymorphism supplies appropriate behavior through a common type.

Why each choice is right or wrong

A - Incorrect: Abstraction concerns the essential view presented to callers, rather than the specific grouping/access boundary requested.

B - Incorrect: Inheritance forms a subtype relationship rather than a state-access boundary.

C - Incorrect: Polymorphism provides behavior through common types rather than defining this grouping and access control.

D - Correct: Encapsulation specifically groups state with operations and controls client access.

Primary note proof

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller. These ideas overlap: a private balance plus deposit(amount) can demonstrate all three, depending on what the question emphasizes.

Study-note section OOP02

> Inheritance forms a subtype relationship and may reuse implementation. Polymorphism lets code using a common type invoke behavior appropriate to different concrete objects.

Question sources

Computer Science Exit Exam · 2018 · Question 43

Question 234 | NS0335

Consider the following code snippet
class Animal { void sound() { System.out.println("Animal sound"); } }
class Dog extends Animal { void sound() { System.out.println("Bark!"); } }
Which OOP concept is illustrated when Dog provides its own version of sound?

A. Method overriding

B. Method overloading

C. Abstraction

D. Encapsulation

Hint

Check inheritance, signature compatibility, visibility and eligibility for an override.

Answer A: Method overriding.

Dog extends Animal and declares its own sound() with the same name, no parameters and the same void result. Neither method is static, private or final in the shown code. Dog therefore overrides the inherited instance method, providing subtype-specific behavior. No changed parameter list exists to make this an overload.

Why each choice is right or wrong

A - Correct: Dog supplies the same eligible inherited sound() signature with its own behavior, which is overriding.

B - Incorrect: Overloading requires a different parameter list; both sound methods have the same no-argument signature.

C - Incorrect: Abstraction is broader essential-interface design, not the specific same-signature subclass mechanism shown.

D - Incorrect: Encapsulation concerns grouped state/access control rather than this replacement of inherited method behavior.

Primary note proof

Study-note section OOP14

> A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype. Visibility cannot become more restrictive, and checked exceptions cannot be broadened incompatibly. @Override requests compiler checking; Java has no override keyword. Constructors, final methods, static methods and private parent methods are not overridden.

Study-note section OOP12

> Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++.

Study-note section OOP02

Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller.

Question sources

Computer Science Exit Exam · 2018 · Question 65

Question 235 | NS0338

A member declared in package p must be usable by an unrelated helper in p and by a subclass in package q through its inherited context. Unrelated classes outside p must be excluded. Which access category fits these requirements?

A. Public

B. Default

C. Private

D. Protected

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer D: Protected, with qualified subclass access across packages.

Protected permits same-package access, including an unrelated helper, and qualified subclass access from another package. It does not grant access to unrelated clients in other packages. Public violates that exclusion, package/default access cannot reach the other-package subclass, and private cannot admit the unrelated same-package helper. For an instance member, the cross-package receiver restriction remains part of the protected rule.

Why each choice is right or wrong

A - Incorrect: Public would also admit the unrelated other-package clients that the stem excludes.

B - Incorrect: Default/package access would exclude the subclass in q.

C - Incorrect: Private would exclude the unrelated helper in p.

D - Correct: Protected provides both requested contexts while retaining the stated other-package exclusion.

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP08

> Across packages, a subclass accessing a protected instance member must do so through its own inherited context or an appropriate subclass-typed receiver, not any arbitrary superclass object. protected therefore does not mean “subclasses only”: unrelated classes in the same package also have access.

Question sources

Computer Science Exit Exam · 2018 · Question 69

Question 236 | NS0345

Consider these Java 17 classes:
class Employee {
public final double calculateTax() { return 100.0; }
}
class Manager extends Employee {
// Insert exactly one candidate declaration here.
}
Which candidate fails compilation because it attempts to override the inherited final method? Evaluate each candidate independently.

A. public double calculateTax(double income) { return income * 0.1; }

B. public double calculateTax() { return 200.0; }

C. private double calculateTax(int category) { return category * 10.0; }

D. public static double calculateTax(String band) { return 0.0; }

Hint

Check inheritance, signature compatibility, visibility and eligibility for an override.

Answer B: public double calculateTax() { return 200.0; }

Employee's no-argument calculateTax method is final, so Manager cannot provide another instance method with that same signature. B attempts exactly that override and fails compilation. A and C have different parameter types and therefore declare overloads. D also has a different parameter list; its static declaration is a separate overload, not a replacement for the inherited no-argument instance method. The final restriction does not prohibit every reuse of the name.

Why each choice is right or wrong

A - Incorrect: The double parameter creates a different signature, so this overload does not override the final no-argument method.

B - Correct: This has the inherited no-argument signature and attempts to override the final method.

C - Incorrect: The int parameter creates an independent overload; its private access does not reduce the inherited method's visibility.

D - Incorrect: The String parameter creates a distinct static overload and leaves the inherited no-argument method unchanged.

Primary note proof

Study-note section OOP14

> Constructors, final methods, static methods and private parent methods are not overridden.

Study-note section OOP12

> Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++.

Study-note section OOP07

> final prevents reassignment of a variable, overriding of a method or extension of a class.

Question sources

Computer Science Exit Exam · 2018 · Question 78

Question 237 | NS0632

A class has a constructor: public Circle(double r) { this.radius = r; }. What happens when you write: Circle c = new Circle(); ?

Hint

Check the class name, return-type absence and the constructors actually declared.

Answer: Compilation fails unless Circle also declares an accessible no-argument constructor.

Declaring Circle(double r) prevents Java from supplying the implicit default constructor. new Circle() requests a no-argument constructor, not the existing one-argument signature. OOP09 therefore predicts a compilation error unless another accessible no-argument constructor has been provided. Passing a double argument would instead target the shown constructor.

Primary note proof

Study-note section OOP09

> If a class declares no constructor, Java supplies a default constructor. If Circle(double r) is declared, new Circle() fails unless a no-argument constructor is also declared.

Question sources

Object-Oriented Programming review questions · Question 1

Question 238 | NS0633

Which of the following is TRUE about instance variables that are not explicitly initialized?

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer: Uninitialized instance fields receive type defaults: numeric zero, false, null, or the zero char.

Instance variables are fields, so OOP05 gives them automatic defaults before explicit initialization. Numeric fields receive zero, boolean fields false, references null and char the zero code unit. This does not authorize reading an unassigned local variable: locals obey a separate definite-assignment requirement rather than receiving these field defaults.

Primary note proof

Study-note section OOP05

> Java fields and array elements receive defaults: numeric zero, char '\u0000', boolean false and references null. A local int t; followed by printing t is a compilation error.

Question sources

Object-Oriented Programming review questions · Question 2

Question 239 | NS0634

What is the output of: static int count = 0; in a class, after creating 3 objects each incrementing count++ in their constructor?

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer: The shared count becomes 3; printing it would output 3.

static makes count a single field shared by the instances, not three independent counters. Starting at zero, each of the three constructor increments adds one to that same field, leaving three. The declaration itself prints nothing; 3 is the stored value and the result if the question's output means subsequently printing count.

Primary note proof

Study-note section OOP07

> static gives one field per loaded class, shared by its instances. If count starts at zero and every constructor increments it, three creations leave count=3. Prefer ClassName.count and ClassName.utility().

Study-note section OOP-S02

> A print call emits its value without inserting a separator or ending the line.

Question sources

Object-Oriented Programming review questions · Question 3

Question 240 | NS0635

You declare: int number; inside a method and try to print it without assigning a value. What happens?

Hint

Locate the declaration scope and check default initialization versus definite assignment.

Answer: Compilation fails because the local variable has not been definitely assigned.

number is declared inside a method, making it a local rather than an instance or static field. OOP05 requires a local to be definitely assigned before use. Printing it is a read, so the compiler rejects that read. Java does not substitute field-style zero or permit an arbitrary garbage output for the local.

Primary note proof

Study-note section OOP05

> Locals occur in methods, constructors or blocks; they must be definitely assigned before use and cannot have public/private/protected modifiers.

Study-note section OOP05

> Java fields and array elements receive defaults: numeric zero, char '\u0000', boolean false and references null. A local int t; followed by printing t is a compilation error.

Question sources

Object-Oriented Programming review questions · Question 4

Question 241 | NS0636

In UML class diagrams, how is an instance shown differently from a class?

Hint

Read the notation and distinguish a class description from a particular instance.

Answer: An instance uses an underlined instanceName:ClassName label, often with slot values.

OOP04 gives an underlined label such as sara:Student for one particular object. A class box describes the common type with its name, attributes and operations; an instance notation can show actual slot values. Underlining static class features is a separate UML convention, so the full instanceName:ClassName pattern identifies the object here.

Primary note proof

Study-note section OOP04

> A UML class box normally contains name, attributes and operations. Visibility uses + public, - private and # protected. A field can be written name:String and an operation getName():String.

Study-note section OOP04

> An instance is conventionally labeled with an underlined instanceName:ClassName, such as sara:Student. Class notation describes a type; instance notation describes one particular object and may show its slot values. Underlining static class features has a different purpose; read the whole notation rather than assuming every underline means the same thing.

Question sources

Object-Oriented Programming review questions · Question 5

Question 242 | NS0637

Under Java 17, what is the key restriction on an explicit this(...) invocation inside a constructor?

Hint

Trace constructor delegation and initialization in the specified Java version.

Answer: It must be the first constructor statement and delegates to another constructor in the same class.

Under Java 17, an explicit this(...) invocation must be the first constructor statement and delegates to another constructor of the same class. The constructor cannot also explicitly invoke super(...); the delegation chain reaches superclass construction through another constructor. The version qualification matters because later Java versions permit a restricted prologue, so the Java 17 rule should not be presented as timeless.

Primary note proof

Study-note section OOP10

> Under Java 17 rules, an explicit this(args) or super(args) invocation must be the first constructor statement. this invokes another constructor in the same class; super invokes the immediate superclass constructor. A constructor cannot explicitly invoke both. If there is no explicit delegation, an implicit super() is inserted; a matching accessible no-argument parent constructor must exist.

Study-note section OOP10

> Java 25 permits restricted statements before explicit constructor invocation; the older first-statement rule must be version qualified.

Question sources

Object-Oriented Programming review questions · Question 6

Question 243 | NS0638

A class has: public static final int MAX = 100; Can a subclass change this value?

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer: No. final prevents assigning a new value to that MAX field.

MAX is one static field whose initialized value is 100. final prevents a subclass or other client from assigning a replacement value to that field. A subclass could declare a separate same-name hidden field, but that would not change the inherited constant. Public controls accessibility, not permission to reassign a final field.

Primary note proof

Study-note section OOP07

> final prevents reassignment of a variable, overriding of a method or extension of a class.

Study-note section OOP16

> Fields are hidden rather than dynamically overridden, and same-signature static methods hide rather than override.

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Question sources

Object-Oriented Programming review questions · Question 7

Question 244 | NS0639

Which access modifier allows access ONLY within the same package (not even subclasses in other packages)?

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer: Package-private/default access: omit an access modifier.

OOP08 says an ordinary member with no access modifier grants package access. This admits other code in the same package but does not admit an other-package subclass merely through inheritance. protected would be broader because it adds qualified cross-package subclass access; private would be narrower. The answer is omission, not a default keyword modifier.

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP08

> Across packages, a subclass accessing a protected instance member must do so through its own inherited context or an appropriate subclass-typed receiver, not any arbitrary superclass object. protected therefore does not mean “subclasses only”: unrelated classes in the same package also have access.

Question sources

Object-Oriented Programming review questions · Question 8

Question 245 | NS0640

Can a constructor be declared private? What is a use case?

Hint

Check the class name, return-type absence and the constructors actually declared.

Answer: Yes. A private constructor can support a factory or controlled singleton creation.

OOP09 explicitly permits private constructors. Restricting constructor access prevents arbitrary outside callers from directly constructing the class, while its own permitted code can still instantiate it. A factory can control creation, or a singleton arrangement can expose a controlled instance. Private construction does not mean the class can never create any object internally.

Primary note proof

Study-note section OOP09

> It initializes the new object and can be overloaded or private. A private constructor can support a factory or controlled singleton creation; it does not mean the class can never create an object internally.

Question sources

Object-Oriented Programming review questions · Question 9

Question 246 | NS0641

What is the difference between a.equals(b) and a == b when comparing objects?

Hint

Separate identity, content equality, immutable values and references that may be reassigned.

Answer: == compares reference identity; equals uses the class's equality policy.

For object references, == asks whether they denote the same object. equals invokes a method whose policy may compare content or value; Object.equals remains identity-based unless overridden. String.equals compares character content. Therefore equals does not automatically mean value equality for every class, and two distinct objects may be equal under an appropriate override.

Primary note proof

Study-note section OOP22

> For object references, == tests identity; equals invokes the class's equality policy. Object.equals uses identity unless overridden; String.equals compares character content. Equal objects must have equal hashCode values when that contract is implemented. toString gives a textual representation, not an equality test.

Question sources

Object-Oriented Programming review questions · Question 10

Question 247 | NS0642

In Java, a class extends two classes simultaneously: public class Dog extends Animal, Mammal {}. What is the result?

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer: Compilation fails: a Java class cannot extend two direct superclasses.

OOP13 explicitly prohibits class Dog extends Animal,Mammal. Java class inheritance permits one direct superclass, so the comma-separated extends clause is invalid rather than a runtime ambiguity to resolve. A class can instead implement multiple interfaces for multiple type contracts, and composition can combine separately held collaborators where multiple state/implementation sources are needed.

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent. A class extends one direct class and can implement multiple interfaces. Single, multilevel and hierarchical inheritance mean one parent, a chain, and several children of one parent. Java does not permit class Dog extends Animal,Mammal.

Study-note section OOP13

> Composition represents has-a, such as Car containing Engine.

Question sources

Object-Oriented Programming review questions · Question 11

Question 248 | NS0643

A private method in a superclass is 'overridden' in a subclass with the same name and signature. What actually happens?

Hint

Check inheritance, signature compatibility, visibility and eligibility for an override.

Answer: The child declaration is independent; it does not override the private parent method.

OOP14 expressly treats a same-signature child method matching a private parent method as an independent declaration. The parent's private method is not an eligible inherited method for runtime overriding. Thus equal spelling and parameters alone do not establish polymorphic replacement, and this is not ordinary static-method hiding either. Calls must follow each declaration's own accessibility and context.

Primary note proof

Study-note section OOP14

> A same-signature child method matching a private parent method is an independent declaration, not runtime overriding and not ordinary static-method hiding.

Study-note section OOP14

> Constructors, final methods, static methods and private parent methods are not overridden.

Question sources

Object-Oriented Programming review questions · Question 12

Question 249 | NS0644

Animal a = new Dog(); - What is the declared type and actual type?

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer: Declared type: Animal. Actual/runtime type: Dog.

Animal is the variable’s declared compile-time type, determining which member operations may be selected by source checking. new Dog() creates the object, making Dog its actual runtime class. Assignment to the Animal reference does not turn that object into an Animal-only instance. Eligible overridden calls still use the runtime Dog implementation.

Primary note proof

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Question sources

Object-Oriented Programming review questions · Question 13

Question 250 | NS0645

You have Animal a = new Dog(); and Dog has a method fetch() not in Animal. What happens when you call a.fetch()?

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer: Compilation fails because fetch() is absent from the declared Animal type.

Although the object is a Dog, the reference a has declared type Animal. OOP15 checks member availability using that declared type before considering runtime overriding. Because fetch() exists only in Dog, a.fetch() is rejected. A checked/safe narrowing to a suitable Dog reference could expose Dog-specific members, but the original expression does not perform that narrowing.

Primary note proof

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Study-note section OOP21

> Use if(a instanceof Ferrari) before narrowing when the dynamic type is uncertain.

Question sources

Object-Oriented Programming review questions · Question 14

Question 251 | NS0646

What is dynamic binding (late binding) in Java?

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer: Eligible overridden instance methods are selected at runtime from the actual object class.

The compiler first verifies accessible members and selects the method signature from declared types. Runtime dispatch then selects the most specific eligible override in the object's actual class. For Animal a=new Dog(), a.sound() can therefore invoke Dog.sound(). This rule does not make Dog-only methods compile through Animal, and static methods/fields follow different selection rules.

Primary note proof

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Study-note section OOP15

> To solve dispatch questions: check the declaration and visibility, determine the compile-time signature, then identify the object's runtime class and its most specific override.

Study-note section OOP16

> Fields are hidden rather than dynamically overridden, and same-signature static methods hide rather than override. With Animal a=new Dog(), a.staticMethod() resolves using Animal's compile-time type. Prefer Animal.staticMethod() for clarity.

Question sources

Object-Oriented Programming review questions · Question 15

Question 252 | NS0647

An abstract class has 3 abstract methods. A subclass implements only 2. What is the result?

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer: The subclass must remain abstract unless it provides or inherits the third implementation; a concrete incomplete subclass fails compilation.

Implementing only two of three required methods leaves an abstract obligation. OOP18 requires a concrete subclass to provide or inherit implementations for every required method. If the third is still unimplemented, declaring the subclass abstract is permitted; declaring it concrete is not. Merely implementing a majority of the methods does not satisfy the parent contract.

Primary note proof

Study-note section OOP18

> A concrete subclass must provide or inherit implementations of all required abstract methods; otherwise it must itself be abstract.

Question sources

Object-Oriented Programming review questions · Question 16

Question 253 | NS0648

Can an abstract class have a constructor? Can it be instantiated?

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer: Yes, it can have constructors; no, it cannot be directly instantiated.

OOP18 allows an abstract class to define constructors alongside fields and concrete/abstract methods. Its constructor participates in initializing the inherited part of a concrete subclass object. That initialization role does not permit new of the abstract class itself. Direct instantiation remains prohibited even if the abstract class happens to contain no abstract methods.

Primary note proof

Study-note section OOP18

> An abstract class cannot be directly instantiated but can define fields, constructors, abstract methods and concrete methods. It need not contain an abstract method.

Study-note section OOP10

> Under Java 17 rules, an explicit this(args) or super(args) invocation must be the first constructor statement. this invokes another constructor in the same class; super invokes the immediate superclass constructor. A constructor cannot explicitly invoke both.

Question sources

Object-Oriented Programming review questions · Question 17

Question 254 | NS0649

What is the difference between implementing an interface and extending a class?

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer: extends forms a single superclass relationship; implements fulfills one or more interface contracts.

A class can extend one superclass, gaining a subtype relationship and appropriate inherited implementation/state. It can also implement multiple interfaces, satisfying their required public operations through provided or inherited methods. Interfaces supply capability contracts rather than per-instance fields or constructors. Modern default/static interface methods mean the distinction cannot simply be summarized as interfaces never contain bodies.

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent. A class extends one direct class and can implement multiple interfaces.

Study-note section OOP19

> A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces. No interface instance fields or constructors exist; fields are implicitly public static final.

Study-note section OOP19

> A method declaration without private, default or static is implicitly public abstract. Modern interfaces may also have default/static method bodies, and Java 9+ private helper methods. Not every interface method must lack a body in modern Java. Implementations must satisfy all inherited abstract contracts; they may inherit suitable public implementations.

Question sources

Object-Oriented Programming review questions · Question 18

Question 255 | NS0650

For an ordinary bodyless Java interface method declaration with no private, default or static modifier, which two method modifiers are implicit?

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer: public and abstract.

The specified bodyless interface method is implicitly public and abstract. An implementing concrete class must satisfy the public contract without reducing its visibility. This scoped rule does not describe every modern interface method: default and static methods may have bodies, and Java 9 and later allow private helper methods. Those forms are explicitly outside the declaration being classified.

Primary note proof

Study-note section OOP19

> A method declaration without private, default or static is implicitly public abstract. Modern interfaces may also have default/static method bodies, and Java 9+ private helper methods.

Question sources

Object-Oriented Programming review questions · Question 19

Question 256 | NS0651

Can an interface have instance fields (instance variables)?

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer: No. Interface fields are implicitly public static final, not per-instance variables.

OOP19 explicitly excludes interface instance fields. Any field declared in an interface is a public static final class-level constant under the note's rule, even when some modifiers are not written. Implementing classes may define their own instance fields, but those belong to the implementing objects rather than being instance state supplied by the interface.

Primary note proof

Study-note section OOP19

> No interface instance fields or constructors exist; fields are implicitly public static final.

Question sources

Object-Oriented Programming review questions · Question 20

Question 257 | NS0652

What is the output? Animal a = new Cat(); a.makeNoise(); - Cat overrides makeNoise() to print 'Meow'

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer: Meow, from Cat.makeNoise(), assuming the inherited method is accessible and eligible for overriding.

The declared Animal type allows the makeNoise call under the stated override relationship. The actual object is a Cat, so runtime dispatch selects Cat's overriding implementation and prints Meow. Assignment to a superclass reference does not suppress subclass behavior. The question does not specify an exact print API, so no additional punctuation or newline is invented.

Primary note proof

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Study-note section OOP15

> To solve dispatch questions: check the declaration and visibility, determine the compile-time signature, then identify the object's runtime class and its most specific override.

Question sources

Object-Oriented Programming review questions · Question 21

Question 258 | NS0653

When is downcasting (Dog dog = (Dog) animal;) safe to use?

Hint

Separate compile-time cast compatibility from the object's actual runtime type.

Answer: For a non-null value, when the actual object is Dog or a subtype; test instanceof when uncertain.

An Animal reference may denote a Dog, another subtype or a plain Animal. Downcasting succeeds for an actual Dog-compatible object and otherwise throws ClassCastException. OOP21 recommends instanceof before narrowing when uncertain. Casting null also succeeds and yields null, but using it as an object afterward fails; therefore successful casting and safe dereferencing are distinct questions.

Primary note proof

Study-note section OOP21

> Upcasting a subclass reference to a superclass is implicit and does not change the object. A downcast may pass compile-time type checks yet fail at runtime. Car a=new Car(); Ferrari f=(Ferrari)a; throws ClassCastException before subsequent drive() or brake() calls. It does not turn a plain Car into a Ferrari.

Study-note section OOP21

> Use if(a instanceof Ferrari) before narrowing when the dynamic type is uncertain. null instanceof Ferrari is false; casting null yields null, although dereferencing it fails.

Question sources

Object-Oriented Programming review questions · Question 22

Question 259 | NS0654

What does the final keyword prevent when applied to a class like: final class String {}?

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer: It prevents subclasses from extending that class.

OOP07 says final on a class prohibits extension. Developers therefore cannot declare a subclass of String. This differs from final on a variable, which prevents reassignment, and final on a method, which prevents overriding. A final class is not automatically immutable merely because subclassing is prohibited; String's immutability is an additional property discussed in OOP22.

Primary note proof

Study-note section OOP07

> final prevents reassignment of a variable, overriding of a method or extension of a class. A final reference may still refer to a mutable object.

Study-note section OOP22

> final class String prevents subclassing; final alone does not make arbitrary objects immutable.

Question sources

Object-Oriented Programming review questions · Question 23

Question 260 | NS0655

Can a static method be overridden in Java?

Hint

Distinguish the current instance, parent context, hidden fields and static dispatch.

Answer: No. A same-signature subclass static method hides the parent static method.

OOP16 states that static methods hide rather than override. Selecting a static method through a reference uses the compile-time type, unlike eligible instance-method dispatch based on the object's runtime class. Thus Animal a=new Dog(); a.staticMethod() selects Animal's static declaration. ClassName.method() is the clearer recommended spelling and makes that selection explicit.

Primary note proof

Study-note section OOP16

> Fields are hidden rather than dynamically overridden, and same-signature static methods hide rather than override. With Animal a=new Dog(), a.staticMethod() resolves using Animal's compile-time type. Prefer Animal.staticMethod() for clarity. Java this and super are unavailable as current-instance references in static context.

Question sources

Object-Oriented Programming review questions · Question 24

Question 261 | NS0656

What is the 'is-a' relationship and how does it relate to inheritance?

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer: A subtype is a valid instance of its parent contract: Dog is an Animal.

OOP13 recommends inheritance when a subtype can satisfy its parent's contract, as Dog satisfies Animal. That is the is-a relationship rather than mere code reuse. Car containing Engine is instead has-a composition. Reusing fields or methods alone does not justify declaring a subclass if its behavior cannot honor the parent's promised operations.

Primary note proof

Study-note section OOP13

> Use inheritance when a subtype can satisfy the parent's contract: Dog is an Animal. For shared Pet.name and speak(), put common state/contract in Pet, specialized fetch() in Dog, and appropriate speak() implementations in Dog and Cat. Composition represents has-a, such as Car containing Engine. Reuse alone does not justify an invalid is-a relationship.

Question sources

Object-Oriented Programming review questions · Question 25

Question 262 | NS0657

You call super() in a subclass constructor but the superclass has no no-arg constructor. What happens?

Hint

Trace constructor delegation and initialization in the specified Java version.

Answer: Compilation fails unless an accessible matching no-argument superclass constructor exists.

An explicit super() requests the immediate superclass's no-argument constructor. OOP10 requires a matching accessible parent constructor, so a superclass with only parameterized constructors cannot satisfy that invocation. The subclass must instead call an appropriate super(arguments) or the superclass must provide an accessible no-argument constructor. Java does not synthesize one after another constructor has been declared.

Primary note proof

Study-note section OOP10

> this invokes another constructor in the same class; super invokes the immediate superclass constructor. A constructor cannot explicitly invoke both. If there is no explicit delegation, an implicit super() is inserted; a matching accessible no-argument parent constructor must exist.

Study-note section OOP09

> If a class declares no constructor, Java supplies a default constructor. If Circle(double r) is declared, new Circle() fails unless a no-argument constructor is also declared.

Question sources

Object-Oriented Programming review questions · Question 26

Question 263 | NS0658

What is the purpose of the protected access modifier specifically for inheritance?

Hint

Locate the accessing code, declaring type, package and receiver before applying visibility rules.

Answer: It permits inherited access in subclasses across packages, and also same-package access.

protected supports subclass use of a parent member without making it universally public. OOP08 also grants access to unrelated code in the same package, so it is not subclasses-only. Across packages an instance member must be accessed in the inherited context or through an appropriate subclass-typed receiver, rather than any arbitrary superclass object.

Primary note proof

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP08

> Across packages, a subclass accessing a protected instance member must do so through its own inherited context or an appropriate subclass-typed receiver, not any arbitrary superclass object. protected therefore does not mean “subclasses only”: unrelated classes in the same package also have access.

Question sources

Object-Oriented Programming review questions · Question 27

Question 264 | NS0659

An interface Drawable extends both Shape and Colorable interfaces. A class Circle implements Drawable. What must Circle implement?

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer: A concrete Circle must satisfy all inherited abstract operations from Drawable, Shape and Colorable.

Drawable can extend both interfaces, combining their contracts. Circle implements Drawable and must therefore provide or inherit suitable public implementations of every remaining abstract operation from the full hierarchy. One compatible implementation can satisfy a shared signature; inherited defaults may supply implementations where applicable. If obligations remain unresolved, Circle must be abstract rather than an incomplete concrete class.

Primary note proof

Study-note section OOP19

> A class can extend a superclass and implement zero or more interfaces. An interface can extend several interfaces.

Study-note section OOP19

> Implementations must satisfy all inherited abstract contracts; they may inherit suitable public implementations.

Study-note section OOP20

> If two compatible interfaces merely declare the same abstract signature, one public implementation can satisfy both. If return types or other contracts conflict, additional declarations may still be impossible.

Study-note section OOP18

> A concrete subclass must provide or inherit implementations of all required abstract methods; otherwise it must itself be abstract.

Question sources

Object-Oriented Programming review questions · Question 28

Question 265 | NS0660

What happens if you declare the same method in both an abstract class and the interface it implements?

Hint

Check the kind of inherited method and whether competing contracts require an explicit resolution.

Answer: Compatible declarations can coexist; implementation obligations depend on abstract, concrete and default status.

OOP20 says a concrete inherited class method takes precedence over an interface default. An abstract class declaration can instead leave an implementation obligation for a concrete subclass. A compatible public implementation may satisfy the interface contract, but incompatible signatures/return types may be impossible. Without declaration details, these conditional rules are the answer rather than one unconditional outcome.

Primary note proof

Study-note section OOP20

> A concrete inherited class method takes precedence over an interface default. An inherited abstract class declaration can require a concrete subclass to implement the operation rather than silently adopting a default.

Study-note section OOP20

> Unrelated interfaces providing conflicting defaults require an explicit resolution; interfaces do not remove every diamond ambiguity. If two compatible interfaces merely declare the same abstract signature, one public implementation can satisfy both. If return types or other contracts conflict, additional declarations may still be impossible. The answer to “same method in abstract class and interface” depends on whether each is abstract or implemented and whether their signatures are compatible.

Question sources

Object-Oriented Programming review questions · Question 29

Question 266 | NS0661

Can you instantiate an interface? If not, how do you use it?

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer: Not directly. Use an interface reference holding an implementing object, anonymous implementation or suitable lambda.

OOP19 prohibits direct interface instantiation while allowing a reference of interface type to hold an implementing object. Calls then use the interface contract with the object's implementation. Anonymous class syntax still creates an implementing class instance, and a compatible functional-interface lambda supplies an implementation; neither means constructing a bare interface with no implementation.

Primary note proof

Study-note section OOP19

> Implementations must satisfy all inherited abstract contracts; they may inherit suitable public implementations. Interfaces cannot be instantiated directly, but references may hold implementing objects, anonymous implementations or suitable functional-interface lambdas.

Question sources

Object-Oriented Programming review questions · Question 30

Question 267 | NS0662

What is the difference between throw and throws in Java?

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer: throw actively raises an exception value; throws declares a method's possible checked-exception obligation.

OOP30 contrasts throw new IOException("message") with throws IOException in a method signature. The first performs an action in control flow, skipping the remaining try body; the second declares an obligation that callers must handle or propagate. A throws clause does not itself throw anything, and throwing a checked exception still requires handling or declaration.

Primary note proof

Study-note section OOP30

> throw new IOException("message") actively throws a value; throws IOException in a signature declares a possible checked exception. A caller must handle or propagate checked obligations. try encloses guarded operations; a matching catch handles the thrown type or a subtype. Execution skips the remainder of the try when a throw occurs.

Question sources

Object-Oriented Programming review questions · Question 31

Question 268 | NS0663

What happens if an exception occurs in a try block and there is no matching catch block?

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer: It propagates outward to a matching enclosing handler; if uncaught, its thread terminates.

Without a matching local catch, the exception is not ignored and execution does not resume with the next ordinary statement in the try. OOP30 sends it outward through enclosing calls/handlers. Relevant finally blocks normally execute during that exit under OOP31's limits. If no handler catches it, the uncaught exception terminates its thread, not necessarily every JVM thread.

Primary note proof

Study-note section OOP30

> Execution skips the remainder of the try when a throw occurs.

Study-note section OOP30

> If no catch matches, the exception propagates outward and an uncaught exception terminates its thread.

Study-note section OOP31

> finally normally executes after try and any selected catch, including when leaving by return or propagating an exception. It is not guaranteed if the process/JVM exits or crashes, or control never leaves the try.

Question sources

Object-Oriented Programming review questions · Question 32

Question 269 | NS0664

Is it mandatory to handle an ArithmeticException?

Hint

Separate source errors from runtime exceptions and determine the hierarchy and checked obligation.

Answer: No catch-or-declare obligation applies: ArithmeticException is unchecked.

OOP29 lists ArithmeticException among unchecked exceptions, so Java does not require a surrounding catch or throws declaration merely because it may occur. A program can still catch it when that provides useful recovery. Unchecked means no mandatory compile-time handling obligation, not harmless behavior: an uncaught occurrence propagates and may terminate its thread.

Primary note proof

Study-note section OOP29

> RuntimeException and Error subclasses are unchecked; other throwable classes are checked. IOException, FileNotFoundException and SQLException are standard checked examples. ArithmeticException, NullPointerException, ClassCastException and array bounds exceptions are unchecked.

Study-note section OOP29

> Checked does not mean the exception occurs at compile time: the compiler checks an obligation to catch or declare it, while an actual throw occurs at runtime.

Study-note section OOP30

> If no catch matches, the exception propagates outward and an uncaught exception terminates its thread.

Question sources

Object-Oriented Programming review questions · Question 33

Question 270 | NS0665

What is the order of execution in: try { ... } catch(E e) { ... } finally { ... }?

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer: Run try, then a matching catch if thrown, then normally finally.

If try completes normally, no catch executes and finally follows. If an exception is thrown, the remaining try body is skipped, a matching catch runs if available, and finally normally executes before continuation or outward propagation. finally also normally runs on return. Process/JVM exit, a crash or control never leaving try prevents an unconditional always claim.

Primary note proof

Study-note section OOP30

> Execution skips the remainder of the try when a throw occurs.

Study-note section OOP31

> finally normally executes after try and any selected catch, including when leaving by return or propagating an exception. It is not guaranteed if the process/JVM exits or crashes, or control never leaves the try.

Question sources

Object-Oriented Programming review questions · Question 34

Question 271 | NS0666

You catch a general Exception before a specific IOException. What is the result?

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer: Compilation fails because the IOException catch after Exception is unreachable.

Exception already catches IOException values because IOException is a subtype. The later subtype handler can never be selected, and OOP30 says this ordering prevents compilation rather than merely causing the general handler to win at runtime. Place the specific IOException catch before Exception if both handlers are needed.

Primary note proof

Study-note section OOP30

> Catch blocks are considered in source order. Place FileNotFoundException before IOException and IOException before Exception; a general catch before a subtype catch makes that later handler unreachable and prevents compilation.

Question sources

Object-Oriented Programming review questions · Question 35

Question 272 | NS0667

What does e.printStackTrace() actually print?

Hint

Track pending returns or exceptions and distinguish normal cleanup from process termination.

Answer: Exception information plus recorded call-stack frames for diagnosis.

OOP31 distinguishes getMessage, toString and printStackTrace. The stack trace reports exception information together with recorded call frames, helping locate the call path associated with the failure. It is more diagnostic context than the message alone and does not repair or handle the error by itself. Exact text depends on the exception and recorded frames.

Primary note proof

Study-note section OOP31

> getMessage returns the message, toString normally includes type and message, and printStackTrace shows exception information and recorded call frames for diagnosis. Tests should cover expected success, caught failures and propagation. Exception handling does not replace validation or correct an algorithm’s logic automatically.

Question sources

Object-Oriented Programming review questions · Question 36

Question 273 | NS0668

Can a finally block change the return value of a try block?

Hint

Track pending returns or exceptions and distinguish normal cleanup from process termination.

Answer: Yes. A return from finally overrides the pending try/catch return; avoid it.

A try return expression is evaluated before finally runs. A return statement inside finally can replace that pending return and can even suppress an exception, which OOP31 warns against. Merely changing a local after a primitive return expression was evaluated does not automatically recompute that earlier result. The explicit finally return is the clear override mechanism.

Primary note proof

Study-note section OOP31

> A return expression is evaluated before finally; a finally return overrides the pending return and can suppress an exception, so avoid that pattern.

Question sources

Object-Oriented Programming review questions · Question 37

Question 274 | NS0669

What is the difference between Error and Exception in Java?

Hint

Separate source errors from runtime exceptions and determine the hierarchy and checked obligation.

Answer: They are Throwable branches: Error usually denotes serious conditions; Exception contains checked and unchecked application exceptions.

OOP29 places Error and Exception below Throwable. Error normally signals serious conditions for which routine recovery is inappropriate and its subclasses are unchecked. Exception includes unchecked RuntimeException subclasses and other checked exceptions requiring catch-or-declare handling. It is technically possible to catch Error, so the distinction must not be stated as errors can never be caught.

Primary note proof

Study-note section OOP29

> Throwable is the root of throwable values, with Exception and Error branches. RuntimeException and Error subclasses are unchecked; other throwable classes are checked.

Study-note section OOP29

> Checked does not mean the exception occurs at compile time: the compiler checks an obligation to catch or declare it, while an actual throw occurs at runtime. A custom exception can extend Exception or RuntimeException. Error normally signals serious conditions for which routine recovery is inappropriate; it is not technically impossible to catch an Error.

Question sources

Object-Oriented Programming review questions · Question 38

Question 275 | NS0670

FileNotFoundException is a subclass of IOException. Which catch block executes: catch(IOException e) or catch(FileNotFoundException e)?

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer: For FileNotFoundException, the first applicable catch wins; put it before IOException.

Both handler types can match a FileNotFoundException, so catch order matters. If FileNotFoundException is first, that specific block executes. If IOException is placed first with FileNotFoundException afterward, the latter is unreachable and compilation fails. If only IOException is present, it can handle the subclass. The prompt omits order, so these cases must be distinguished.

Primary note proof

Study-note section OOP30

> Execution skips the remainder of the try when a throw occurs.

Study-note section OOP30

> Catch blocks are considered in source order. Place FileNotFoundException before IOException and IOException before Exception; a general catch before a subtype catch makes that later handler unreachable and prevents compilation. If no catch matches, the exception propagates outward and an uncaught exception terminates its thread.

Question sources

Object-Oriented Programming review questions · Question 39

Question 276 | NS0671

What does ClassCastException indicate?

Hint

Separate compile-time cast compatibility from the object's actual runtime type.

Answer: An attempted reference cast is incompatible with the actual object type.

A cast can be syntactically and type-checkably permitted between related reference types while still fail at runtime. OOP21's plain Car cannot become a Ferrari merely by casting, so ClassCastException occurs before subsequent method calls. The exception reports an incompatible dynamic type, not object transformation, allocation or a numeric conversion failure.

Primary note proof

Study-note section OOP21

> Upcasting a subclass reference to a superclass is implicit and does not change the object. A downcast may pass compile-time type checks yet fail at runtime. Car a=new Car(); Ferrari f=(Ferrari)a; throws ClassCastException before subsequent drive() or brake() calls. It does not turn a plain Car into a Ferrari.

Study-note section OOP21

> For Object obj="Hello", (String)obj succeeds and (Integer)obj throws ClassCastException.

Question sources

Object-Oriented Programming review questions · Question 40

Question 277 | NS0672

Can you use multiple catch blocks for one try block? Can one catch block catch multiple exceptions?

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer: Yes to both: use ordered catches or a multi-catch such as catch(IOException | SQLException e).

One try can have several catch blocks; source order and specific-before-general reachability still apply. A single multi-catch can also handle alternative exception types. OOP30 prohibits alternatives related by subclassing, so IOException and FileNotFoundException cannot be paired in one union catch. Their hierarchy instead calls for an appropriate common catch or ordered handlers.

Primary note proof

Study-note section OOP30

> Catch blocks are considered in source order. Place FileNotFoundException before IOException and IOException before Exception; a general catch before a subtype catch makes that later handler unreachable and prevents compilation. If no catch matches, the exception propagates outward and an uncaught exception terminates its thread. Multiple catch blocks are allowed; catch(IOException | SQLException e) handles alternatives, which must not be related by subclassing.

Question sources

Object-Oriented Programming review questions · Question 41

Question 278 | NS0673

What is the output? int x = 5; int y = ++x * x--; System.out.println(x + ' ' + y);

Hint

Track the old and updated values separately and distinguish a character operand from a string.

Answer: 73. The expression leaves x=5 and y=36, then adds the space character's value 32.

Java evaluates ++x first, changing 5 to 6 and supplying 6. x-- then supplies 6 and restores x to 5, so y=36. In x + ' ' + y the space is char, not String, and numeric promotion gives 5+32+36=73. Replacing it with a double-quoted space would instead concatenate separated values.

Primary note proof

Study-note section OOP24

> Starting x=5, printing x++ then ++x gives 5 then 7. For int y=++x*x--, Java uses 6*6=36 and leaves x=5. System.out.println(x + ' ' + y) then adds char space's numeric value 32, so it prints 73. To print separated values, use x + " " + y instead.

Study-note section OOP06

> Arithmetic promotes char, byte and short to int in many expressions.

Question sources

Object-Oriented Programming review questions · Question 42

Question 279 | NS0674

Circle and Sphere are separately declared top-level Java classes. Circle declares private double radius; and Sphere extends Circle. What happens if an instance method of Sphere attempts this.radius = 5.0;?

Hint

Separate parameters from arguments and operations that read state from operations that change it.

Answer: Compilation fails for an ordinary separately declared Sphere: the inherited radius field is private.

The direct field access fails compilation. Circle's private field is not accessible from a separately declared top-level subclass, and this only identifies the current object; it does not relax visibility. Sphere must use an accessible operation supplied by Circle if it needs to request a radius change. Declaring the classes separately removes the shared-nest exception from the scenario.

Primary note proof

Study-note section OOP11

> A private field cannot normally be accessed directly by a separately declared subclass, even with this.field; use a permitted parent operation.

Study-note section OOP08

> For ordinary members: private restricts access to the declaring class/nest; no modifier grants package access; protected grants same-package access plus qualified subclass access; public is widest, provided the declaring type itself is accessible.

Study-note section OOP16

> Java this denotes the current object; this.name=name distinguishes a field from a parameter.

Question sources

Object-Oriented Programming review questions · Question 43

Question 280 | NS0675

What is the difference between a concrete class and an abstract class?

Hint

Check whether a class is concrete and whether every inherited abstract contract has an implementation.

Answer: A concrete class can be instantiated and must satisfy abstract obligations; an abstract class cannot be directly instantiated.

An abstract class can hold state, constructors, concrete methods and abstract methods, and need not contain an abstract method. A concrete subclass must provide or inherit implementations of every required abstract operation before it can serve as an instantiable class. Abstract is therefore a class-level instantiation/contract restriction, not a prohibition on implemented behavior.

Primary note proof

Study-note section OOP18

> An abstract class cannot be directly instantiated but can define fields, constructors, abstract methods and concrete methods. It need not contain an abstract method.

Study-note section OOP18

> A concrete subclass must provide or inherit implementations of all required abstract methods; otherwise it must itself be abstract.

Question sources

Object-Oriented Programming review questions · Question 44

Question 281 | NS0676

String is declared as a final class. What does this mean for developers?

Hint

Decide whether a member belongs to a class or an object, then check initialization and final restrictions.

Answer: Developers cannot subclass String; final prohibits extension.

Applying final to String prevents any class from extending String. It does not prevent constructing or referencing String instances, nor does the class modifier alone explain immutability of arbitrary objects. OOP22 treats String's immutable content as a distinct property. Developers should use existing String operations rather than attempting a String subclass.

Primary note proof

Study-note section OOP07

> final prevents reassignment of a variable, overriding of a method or extension of a class. A final reference may still refer to a mutable object.

Study-note section OOP22

> Java String objects are immutable. text += " World" assigns a reference to a new concatenated result and can print Hello World without mutating the original String object. final class String prevents subclassing; final alone does not make arbitrary objects immutable.

Question sources

Object-Oriented Programming review questions · Question 45

Question 282 | NS0677

What is method hiding in Java and how does it differ from overriding?

Hint

Distinguish the current instance, parent context, hidden fields and static dispatch.

Answer: A subclass static method hides a same-signature parent static method; overriding is runtime instance dispatch.

For static hiding, the selected declaration follows the compile-time type used to qualify the call. For eligible overridden instance methods, the object's runtime class determines the implementation. OOP16's Animal a=new Dog() static call therefore resolves to Animal, unlike a virtual instance sound call that could invoke Dog. Class qualification makes static selection clearer.

Primary note proof

Study-note section OOP16

> Fields are hidden rather than dynamically overridden, and same-signature static methods hide rather than override. With Animal a=new Dog(), a.staticMethod() resolves using Animal's compile-time type. Prefer Animal.staticMethod() for clarity.

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Question sources

Object-Oriented Programming review questions · Question 46

Question 283 | NS0678

Given: interface A { void m1(); } interface B { void m2(); } class C implements A, B {} - What must C define?

Hint

Identify the interface contract, effective modifiers and permitted implementation forms.

Answer: C must provide public void m1() and public void m2(), or inherit suitable implementations.

Both unmodified interface declarations are implicitly public abstract. A concrete C implementing A and B must satisfy both signatures, not merely one. In the shown class with no suitable superclass implementation, define public void m1() and public void m2(). Package-private implementations would narrow the public contract; otherwise C must be declared abstract instead of concrete.

Primary note proof

Study-note section OOP19

> A method declaration without private, default or static is implicitly public abstract. Modern interfaces may also have default/static method bodies, and Java 9+ private helper methods. Not every interface method must lack a body in modern Java. Implementations must satisfy all inherited abstract contracts; they may inherit suitable public implementations.

Study-note section OOP18

> A concrete subclass must provide or inherit implementations of all required abstract methods; otherwise it must itself be abstract.

Study-note section OOP14

> Visibility cannot become more restrictive, and checked exceptions cannot be broadened incompatibly.

Question sources

Object-Oriented Programming review questions · Question 47

Question 284 | NS0679

What is the significance of the java.lang.Throwable class?

Hint

Separate source errors from runtime exceptions and determine the hierarchy and checked obligation.

Answer: Throwable is the common root of Java's throwable Error and Exception hierarchy.

OOP29 places both Error and Exception under Throwable, establishing the common type for Java throwable values. The distinction between checked and unchecked branches controls catch-or-declare obligations, not whether an exception happens during compilation. Throwable therefore unifies the hierarchy without implying that every subclass is an ordinary recoverable application exception.

Primary note proof

Study-note section OOP29

> Throwable is the root of throwable values, with Exception and Error branches. RuntimeException and Error subclasses are unchecked; other throwable classes are checked.

Study-note section OOP29

> Checked does not mean the exception occurs at compile time: the compiler checks an obligation to catch or declare it, while an actual throw occurs at runtime. A custom exception can extend Exception or RuntimeException. Error normally signals serious conditions for which routine recovery is inappropriate; it is not technically impossible to catch an Error.

Question sources

Object-Oriented Programming review questions · Question 48

Question 285 | NS0680

A method throws a checked exception but neither catches it nor declares throws. What happens?

Hint

Check the thrown type and catch order before tracing propagation or handler output.

Answer: Compilation fails because the checked exception is neither handled nor declared.

OOP30 requires callers and methods to handle or propagate checked obligations. A method that may throw a checked exception but neither catches it nor declares it in throws violates that compile-time requirement. The failure is not that checked exceptions occur during compilation: the compiler is checking a possible future runtime throw.

Primary note proof

Study-note section OOP30

> throw new IOException("message") actively throws a value; throws IOException in a signature declares a possible checked exception. A caller must handle or propagate checked obligations. try encloses guarded operations; a matching catch handles the thrown type or a subtype.

Study-note section OOP29

> Checked does not mean the exception occurs at compile time: the compiler checks an obligation to catch or declare it, while an actual throw occurs at runtime.

Question sources

Object-Oriented Programming review questions · Question 49

Question 286 | NS0681

What does 'extensibility' mean in the context of polymorphism?

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer: New subtypes can supply behavior while existing common-type client code remains reusable.

OOP15 describes a common Animal array loop invoking speak on Dog and Cat objects. Adding another conforming Animal subtype can supply its own override without rewriting that generic loop. That ability is extensibility through polymorphism: clients depend on the shared contract, while runtime dispatch supplies subtype behavior. The new subtype must still satisfy that contract.

Primary note proof

Study-note section OOP15

> To solve dispatch questions: check the declaration and visibility, determine the compile-time signature, then identify the object's runtime class and its most specific override. An array Animal[]{new Dog(),new Cat()} can call speak() on each using one loop. New subclasses can supply behavior without rewriting this generic loop, which is the practical meaning of extensibility.

Question sources

Object-Oriented Programming review questions · Question 50

Question 287 | NS0682

You have: Object obj = 'Hello'; String s = (String) obj; Is this valid? What about Integer i = (Integer) obj;?

Hint

Separate compile-time cast compatibility from the object's actual runtime type.

Answer: Literal single-quoted Hello is invalid. With "Hello", the String cast succeeds and the Integer cast throws ClassCastException.

A Java character literal cannot contain the multi-character text 'Hello', so the printed declaration fails before either cast executes. OOP21 explicitly separates that defect from the intended double-quoted version. After changing it to a String literal, obj refers to a String: narrowing back to String succeeds, but narrowing it to Integer fails at runtime.

Primary note proof

Study-note section OOP21

> For Object obj="Hello", (String)obj succeeds and (Integer)obj throws ClassCastException. A literal 'Hello' is invalid Java syntax; use double quotes for a string literal before tracing the program.

Study-note section OOP-S04

> A double-quoted string literal denotes a String object. new String(existingString) creates a String with the same text; new String(charArray) copies the array's characters into an immutable String. A single-quoted literal denotes one char, so several characters enclosed together in single quotes cannot supply a String.

Question sources

Object-Oriented Programming review questions · Question 51

Question 288 | NS0683

If a subclass constructor doesn't explicitly call super(), what does Java do?

Hint

Trace constructor delegation and initialization in the specified Java version.

Answer: If no explicit this(...) or super(...) delegation exists, Java inserts super(); this(...) instead delegates within the class.

Omitting explicit super() alone is insufficient to determine what happens. A constructor may explicitly call this(...), delegating to another same-class constructor that eventually reaches superclass construction. Only when there is no explicit constructor delegation does OOP10 insert super(), requiring an accessible matching parent no-argument constructor. Otherwise the program fails to compile.

Primary note proof

Study-note section OOP10

> Under Java 17 rules, an explicit this(args) or super(args) invocation must be the first constructor statement. this invokes another constructor in the same class; super invokes the immediate superclass constructor. A constructor cannot explicitly invoke both. If there is no explicit delegation, an implicit super() is inserted; a matching accessible no-argument parent constructor must exist.

Question sources

Object-Oriented Programming review questions · Question 52

Question 289 | NS0734

What is the difference between a class and an object? Use a concrete example.

Hint

Distinguish an object’s stored state, operations, identity and class definition.

Answer: A class defines common structure/behavior; an object is one instance, such as a particular Rectangle with width and height.

Rectangle can define width, height, getArea and getPerimeter for every rectangle. A particular rectangle object stores its own width and height values and responds to those operations. OOP01 distinguishes that instance from the type description: two objects can have equal field values and still have distinct identities. The class is shared structure, not one particular rectangle.

Primary note proof

Study-note section OOP01

> An object has identity, state and behavior. State is stored in fields; behavior is provided by methods. A Rectangle object may hold width and height and answer getArea() or getPerimeter(). Two objects can have equal fields and still be distinct objects. A class defines their common structure and behavior; an object is a particular instance.

Question sources

Object-Oriented Programming review questions · Question 1

Question 290 | NS0735

Why is encapsulation important? How is it achieved?

Hint

Identify whether the emphasis is grouping state, hiding implementation, exposing essentials or varying behavior.

Answer: Encapsulation protects invariants and stable interfaces by grouping state with validated operations and controlling access.

OOP02 groups state and operations with controlled access, keeping representation decisions behind stable interfaces. A private account balance exposed through deposit(amount) can reject invalid negative changes rather than permitting unrestricted assignment. This localizes responsibility and lets clients use meaningful operations without depending on internal representation. OOP11 cautions that automatically exposing setters for every field is not required.

Primary note proof

Study-note section OOP02

> Encapsulation groups related state and operations and controls access. Information hiding keeps implementation decisions behind a stable interface. Abstraction presents the essential operations while omitting details irrelevant to the caller. These ideas overlap: a private balance plus deposit(amount) can demonstrate all three, depending on what the question emphasizes.

Study-note section OOP11

> An accessor/getter returns information; a mutator/setter changes state. Encapsulation does not require blindly generating setters for every field: validate changes and expose only useful operations. For a bank account, deposit must reject negative amounts.

Question sources

Object-Oriented Programming review questions · Question 2

Question 291 | NS0736

What is the difference between method overloading and method overriding? Provide examples.

Hint

Compare names and parameter signatures before considering return types or access modifiers.

Answer: Overloading changes parameter signatures; overriding replaces eligible inherited instance behavior with the same signature.

For overloading, f(int), f(double) and f(int,int) share a name but have distinct signatures selected using declared arguments at compile time. For overriding, Dog extends Animal and supplies void sound() matching Animal's eligible void sound(); a superclass reference can then dispatch to Dog at runtime. Changing only the return type cannot create an overload, and override return compatibility still matters.

Primary note proof

Study-note section OOP12

> Overloaded methods share a name but have different parameter types, counts or type order. Return type alone is insufficient in Java and C++. Overloads can also arise through inherited methods; "same class only" is a teaching simplification. f(int), f(double) and f(int,int) are distinct, but changing a parameter name does not create an overload.

Study-note section OOP12

> The compiler selects the applicable signature from the declared argument types and conversions.

Study-note section OOP14

> A subclass overrides an eligible inherited instance method with the same signature and a compatible return type; a covariant reference return may be a subtype. Visibility cannot become more restrictive, and checked exceptions cannot be broadened incompatibly.

Study-note section OOP15

> For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Question sources

Object-Oriented Programming review questions · Question 3

Question 292 | NS0737

Why can't Java classes inherit from multiple parent classes? How is this limitation overcome?

Hint

Identify the subtype or ownership relationship and distinguish class inheritance from interface contracts.

Answer: Java permits one direct superclass. Use multiple interfaces for type contracts and composition for cooperating implementation/state.

The immediate reason the declaration is rejected is Java's one-direct-superclass rule in OOP13. Multiple interfaces supply multiple type contracts, while contained collaborators model has-a reuse without multiple class inheritance. This does not eliminate every diamond problem: OOP20 requires explicit resolution of conflicting unrelated interface defaults. The note supplies these mechanisms, rather than a historical language-design rationale.

Primary note proof

Study-note section OOP13

> class Dog extends Animal declares a subclass/derived/child of a superclass/base/parent. A class extends one direct class and can implement multiple interfaces. Single, multilevel and hierarchical inheritance mean one parent, a chain, and several children of one parent. Java does not permit class Dog extends Animal,Mammal.

Study-note section OOP13

> Composition represents has-a, such as Car containing Engine. Reuse alone does not justify an invalid is-a relationship.

Study-note section OOP20

> Unrelated interfaces providing conflicting defaults require an explicit resolution; interfaces do not remove every diamond ambiguity.

Question sources

Object-Oriented Programming review questions · Question 4

Question 293 | NS0738

What is polymorphism? Explain with a code example.

Hint

Check what compiles from the declared type, then select behavior from the actual runtime class.

Answer: Polymorphism lets a common-type reference invoke behavior appropriate to the actual subtype.

Example: class Animal { void sound() { System.out.println("Animal"); } } class Dog extends Animal { void sound() { System.out.println("Bark"); } } Then Animal a=new Dog(); a.sound(); prints Bark. The declared Animal type makes sound available; the actual Dog object supplies its eligible override at runtime. A new Animal subtype can participate through the same common operation.

Primary note proof

Study-note section OOP15

> For Animal a=new Dog(), Animal is the declared type and Dog the actual type. The declared type determines which member operations compile. If fetch() exists only in Dog, a.fetch() fails compilation. For an eligible overridden sound(), a.sound() invokes Dog.sound() at runtime.

Study-note section OOP15

> To solve dispatch questions: check the declaration and visibility, determine the compile-time signature, then identify the object's runtime class and its most specific override. An array Animal[]{new Dog(),new Cat()} can call speak() on each using one loop. New subclasses can supply behavior without rewriting this generic loop, which is the practical meaning of extensibility.

Question sources

Object-Oriented Programming review questions · Question 5

Question 294 | NS0739

When would you use an abstract class versus an interface?

Hint

Check the kind of inherited method and whether competing contracts require an explicit resolution.

Answer: Use an abstract class for shared state/construction/implementation; use an interface for capabilities across unrelated classes and multiple type contracts.

An abstract class can centralize common fields, constructor initialization and concrete operations while leaving selected operations abstract. An interface specifies a capability that otherwise unrelated classes can implement, and a class may implement several interfaces. Modern interface defaults do not supply per-instance fields or constructors, so shared object state/construction remains a reason to choose an abstract class.

Primary note proof

Study-note section OOP20

> Use an abstract class for shared state, construction and common implementation; use an interface for a capability across unrelated classes and multiple type inheritance.

Study-note section OOP19

> No interface instance fields or constructors exist; fields are implicitly public static final.

Study-note section OOP18

> An abstract class cannot be directly instantiated but can define fields, constructors, abstract methods and concrete methods.

Question sources

Object-Oriented Programming review questions · Question 6

Computer Organization and Architecture

Questions & Worked Answers

184 questions

Question 1 | Q00029

Which one of the following is responsible for controlling the flow of bits between memory and controllers with no need for a constant intervention from the CPU?

A. Disk fragmenting

B. Memo, manager

C. Device controller

D. Direct memory access

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: Direct memory access

DMA transfers a block directly between device and memory; the CPU supplies setup information and receives completion notification, rather than moving every word (PDF pp.107-108). This precisely meets the absence of constant intervention. A generic device controller alone does not establish DMA capability.

Why each choice is right or wrong

A - Incorrect: Disk fragmenting is not the block-transfer mechanism described on pp.107-108.

B - Incorrect: The corrupted name “Memo, manager” does not identify the DMA mechanism on p.107.

C - Incorrect: An I/O controller can notify the CPU under interrupt I/O; that still leaves CPU data movement (p.106).

D - Correct: Direct transfers and only beginning/end CPU involvement are explicit on p.108.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

DMA lets a controller move a block directly between device and memory after CPU setup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

7.10 Direct Memory Access (DMA)
Interrupt driven and programmed I/O require active CPU intervention.
Transfer rate is limited
CPU is tied up DMA is the solution for these problems
Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
• Whether to read or write
• Device address
• Starting address of memory block for data
• Amount of data to be transferred The CPU carries on with other work.
• Thus DMA controller steals the CPU‟s work of I/O operation.
• The DMA module transfers the entire block of data
• One word at a time, directly to or from memory, without going through CPU.
When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Question sources

MOE Computer Science Exit Exam · 2016 · Question 29

Question 2 | Q00044

Which one of the following is a communication pathway connecting transmitting signals in binary digits between two or more devices?

A. Memory

B. Register

C. CPU

D. Bus

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: Bus

The common bus is a set of shared lines used to transfer binary information among registers (PDF p.40). The structural definitions distinguish storage and processing components from their communication mechanism (p.6).

Why each choice is right or wrong

A - Incorrect: Main memory stores data; it is an endpoint (p.6).

B - Incorrect: Registers store bits, rather than naming the pathway (p.35).

C - Incorrect: CPU processes and controls; it is a communicating component (p.6).

D - Correct: A bus supplies common bit lines and selection controls (p.40).

Primary note proof

Study-note section COA-S02

A bus is a set of common signal lines used to transfer binary information among connected units.

Study-note section COA01

The CPU contains an ALU, control unit and registers; a microcontroller is a complete integrated control system, not another mandatory CPU subunit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 40 | lines 18-23

4.2 Bus and Memory Transfers
A typical computer has many registers, and paths must be provided to transfer information from one register to another. An efficient way for transferring information between registers in a multiple-register configuration is a common bus system. A bus structure consists of a set of common lines, one for each bit of a register. Control signals determine which register the bus selects during each particular register transfer.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

3.9 Registers
• A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
• A register is a digital circuit used within the CPU to store one or more bits of data.
• An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
• In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
• Two basic types of registers are commonly used: parallel registers and shift registers.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 44

Question 3 | Q00056

Which memory is characterized by the need for periodic change in order to maintain data storage?

A. Static RAM

B. Dynamic RAM

C. Disk

D. Tape

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Dynamic RAM

Interpret “periodic change” as periodic charge refreshing: capacitor charge leaks, so DRAM needs refresh (PDF p.80). This is different from SRAM power retention and nonvolatile secondary storage (p.91).

Why each choice is right or wrong

A - Incorrect: SRAM uses flip-flops and holds data while powered (p.80), without the stated charge refresh.

B - Correct: DRAM capacitor discharge explicitly requires periodic refreshing (p.80).

C - Incorrect: Disk is nonvolatile secondary storage (p.91), not capacitor-refresh RAM.

D - Incorrect: Tape is also nonvolatile secondary storage (p.91), not refresh RAM.

Primary note proof

Study-note section COA24

> SRAM uses powered bistable storage: fast and costly per bit, without periodic refresh.

Study-note section COA24

> DRAM stores charge and requires refresh: denser and commonly used for main memory.

Study-note section COA28

> It does not need continuous power to preserve stored bits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

> DYNAMIC RAM (DRAM)
> is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.

STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 56

Question 4 | Q00060

In the ordinary HDD model, which statement is incorrect? Read/write refers to accessing sectors on the current track; seeking to another track is a separate operation.

A. Data is arranged in sectors within tracks

B. The head stays over the selected track while its sectors rotate beneath it

C. Its data is organized in tracks

D. It needs continuous power to keep bit contents

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: It needs continuous power to keep bit contents

An HDD records magnetic patterns in sectors arranged on tracks. During access to a selected track, rotation brings sectors under the head; seeking moves the head radially to another track. The medium retains its magnetic contents without continuous power, so the claimed power requirement is false. Power is needed to operate the drive, but that differs from power needed merely to retain the stored bits.

Why each choice is right or wrong

A - Incorrect: Sectors are subdivisions of tracks in the stated HDD organization.

B - Incorrect: This describes track-local access; a separate seek changes which track is under the head.

C - Incorrect: Magnetic disk data is organized on tracks, so this statement is true.

D - Correct: An HDD is nonvolatile: retaining stored patterns does not require continuous power.

Primary note proof

Study-note section COA28

An HDD stores magnetic patterns on tracks divided into sectors; reads/writes access addressed blocks while platters rotate.

Study-note section COA28

A head normally stays over a track while reading but moves radially for seeks.

Study-note section COA28

It does not need continuous power to preserve stored bits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 60

Question 5 | Q00093

Which listed circuit category requires stored state so that output can depend on past input behavior as well as present inputs?

A. Combinational circuit

B. Sequential circuits

C. A two-input AND gate with no feedback

D. A combinational multiplexer with no storage

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer B: Sequential circuits

A sequential circuit contains state that preserves information about prior behavior. A combinational circuit computes its outputs from current inputs after propagation delay. The specified AND gate and multiplexer are combinational examples without storage or feedback. They therefore do not require remembered state, while the sequential-circuit category does. This distinction concerns logical state, not whether physical propagation takes time.

Why each choice is right or wrong

A - Incorrect: Combinational output depends on current inputs rather than stored history.

B - Correct: Sequential circuits retain state and can make output depend on past behavior.

C - Incorrect: The specified AND gate computes a truth function of its present inputs without storage.

D - Incorrect: The specified multiplexer selects a present data input; it has no stored logical state.

Primary note proof

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Study-note section COA05

> Adders, multiplexers, demultiplexers, encoders and decoders have no stored logical state.

Study-note section COA05

> A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Study-note section COA03

> AND is 1 only when all inputs are 1; OR is 1 if any input is 1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

• In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
• However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
• Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
• Such circuits include storage elements that store the values of logic signals.
• Flip-flops can store a one-bit data (information).

Computer Organization and Architecture module | PDF page 10 | lines 22-27

1.4 Combinational Circuits
• A combinational circuit is an interconnected set of gates whose output at any time is a function only of the input at that time. As with a single gate, the appearance of the input is followed almost immediately by the appearance of the output, with only gate delays. In general terms, a combinational circuit consists of n binary inputs and m binary outputs.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 93

Question 6 | Q00134

In computers, subtraction is carried out generally by _____.

A. 1's complement method

B. 2's complement method

C. Signed magnitude method

D. BCD subtraction method

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer B: 2's complement method

Subtraction is described as complementation plus addition: add one to the inverted subtrahend, then add that two’s complement to the minuend (PDF p.42). Hence R1 + (~R2+1) represents R1−R2 within the register width.

Why each choice is right or wrong

A - Incorrect: One’s complement alone omits the added 1 in p.42’s subtraction procedure.

B - Correct: The inversion-plus-one subtrahend is exactly two’s complement (p.42).

C - Incorrect: The cited procedure does not use separate sign and magnitude arithmetic.

D - Incorrect: The cited general register subtraction is not BCD digit subtraction.

Primary note proof

Study-note section COA09

> Compute A−B as A+(two’s complement of B), retaining n result bits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 42 | lines 21-25

> Substruction is most often implemented through complementation and addition.
> Example, ¯
> ¯is the symbol for the 1’s complement of R2. Adding 1 to the 1’s complement produces the 2’s complement. Adding the contents of R1 to the 2’s complement of R2 is equivalent to R1-R2.

Question sources

Addis Ababa University mock exam · Question 36

Computer Science Exit Exam · Sene 2016 · Question 20-3

Computer Science Exit Exam practice collection · Question 239

Question 7 | Q00135

Pick the different _

A. Decoder

B. Encoder

C. Multiplexer

D. Shift Register

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer D: Shift register: the sequential circuit.

COA05 distinguishes combinational circuits from sequential storage. Decoder, encoder and multiplexer are explicitly combinational; the shift register retains state and therefore belongs to the sequential class. The image confirms the four option texts. Using the state criterion provides the technically supported odd-one-out distinction without relying on the source emphasis.

Why each choice is right or wrong

A - Incorrect: A decoder is combinational and has no stored logical state in this circuit classification.

B - Incorrect: An encoder is also combinational, grouping it with the decoder and multiplexer.

C - Incorrect: A multiplexer selects among present inputs and is explicitly a combinational circuit.

D - Correct: A shift register retains and shifts stored state, distinguishing it from the three combinational devices.

Primary note proof

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Study-note section COA05

> A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 33 | lines 7-14

> 3.7 Decoders
> • A decoder is a combinational circuit with a number of output lines, only one of which is selected at any time, depending on the pattern of input lines.
> • In general, a decoder has n inputs and 2n outputs.

• Decoders find many uses in digital computers.
• One example is address decoding.
• The other is binary-to- octal conversion.

Computer Organization and Architecture module | PDF page 34 | lines 3-8

3.8 Encoders
• An encoder is a digital circuit that performs the inverse operation of a decoder.
• An encoder has 2n
(or less) input lines and n output lines. The output lines generate
the binary code corresponding to the input value.
• An example of an encoder is the octal-to-binary encoder.

Computer Organization and Architecture module | PDF page 31 | lines 12-16

3.5 Multiplexers
• Multiplexer is a combinational circuit that receives binary information from one of 2n input data & directs to one output.
• The multiplexer connects multiple inputs to a single output.
• At any time, one of the inputs is selected to be passed to the output.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

3.9 Registers
• A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
• A register is a digital circuit used within the CPU to store one or more bits of data.
• An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
• In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
• Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

• In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
• However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
• Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
• Such circuits include storage elements that store the values of logic signals.
• Flip-flops can store a one-bit data (information).

Computer Organization and Architecture module | PDF page 36 | lines 2-9

Shift Register
• A shift register accepts and/or transfers information serially.
• Data are input only to the leftmost flip-flop.
• Shift registers can be used to interface to serial I/O devices.
• With each clock pulse, data are shifted to the right, one position, and the rightmost bit

is transferred out.
• In addition, they can be used within the ALU to perform logical shift and rotate functions.

Question sources

Addis Ababa University mock exam · Question 37

Question 8 | Q00136

An operation performed on the contents of a register is known as

A. Instruction code

B. Micro-operation

C. Accumulator

D. Register

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Micro-operation

An elementary operation on information stored in one or more registers is explicitly defined as a micro operation (PDF p.38). An instruction code tells the computer what operation to execute, while registers are the storage objects (pp.35,47).

Why each choice is right or wrong

A - Incorrect: An instruction code specifies work; it is not the elementary register operation itself (p.47).

B - Correct: Matches the exact definition and examples shift, count, clear, load (p.38).

C - Incorrect: Accumulator denotes a processing register, not the operation.

D - Incorrect: A register stores the information being operated on (p.35).

Primary note proof

Study-note section COA11

A microoperation is one elementary operation on register contents.

Study-note section COA16

An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 38 | lines 8-12

Digital modules are best defined by the registers they contain and the operations that are performed on the data stored in them. The operations executed on data stored in registers are called micro operations. A micro operation is an elementary operation performed on the information stored in one or more registers.
Examples of micro operation are shift, count, clear, and load.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

3.9 Registers
• A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
• A register is a digital circuit used within the CPU to store one or more bits of data.
• An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
• In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
• Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Question sources

Addis Ababa University mock exam · Question 38

Question 9 | Q00137

Which one is true about RISC computers?

A. RISC are with few numbers of registers when compared to CISC

B. RISC use complex and efficient machine instructions when compared to CISC

C. RISC is with hardwired control unit with pipelining processing

D. RISC uses extensive addressing capabilities for memory operations

E. None

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer C: RISC is with hardwired control unit with pipelining processing

RISC characteristics explicitly include hardwired control, many registers, few addressing modes and an efficient pipeline (PDF p.77). These jointly support C and contradict the reversed CISC-like descriptions.

Why each choice is right or wrong

A - Incorrect: A reverses the relatively large register count on p.77.

B - Incorrect: Complex instructions are the CISC characteristic (p.78), whereas RISC has few, easily decoded instructions (p.77).

C - Correct: Both hardwired control and efficient pipeline are explicitly listed (p.77).

D - Incorrect: RISC has relatively few addressing modes and memory access limited to loads/stores (p.77).

E - Incorrect: C is supported, so “None” fails.

Primary note proof

Study-note section COA-S04

> The textbook RISC design pairs a relatively large register set, load/store access and easily decoded instructions with hardwired rather than microprogrammed control and an efficient pipeline.

Study-note section COA20

> CISC commonly supplies more complex operations, richer addressing modes and variable lengths.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 78 | lines 2-9

5.19 CISC Instruction:
CISC refers to Complex Instruction Set Computer
CISC characteristic
• Large number of instructions - typically from 100 to 250 instructions
• Some instructions that perform specialized tasks and are used infrequently
• A large variety of addressing modes - typically from 5 to 20 different modes
• Variable length instruction formats
• Instructions that manipulate operands in memory RISC characteristics

Question sources

Addis Ababa University mock exam · Question 39

Question 10 | Q00138

Which of the following is not a basic element within the microprocessor?

A. Microcontroller

B. Arithmetic logic unit (ALU)

C. Register array

D. Control unit

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Microcontroller

The module divides the CPU into the register set, ALU and control unit (PDF p.57). Treating microprocessor as the CPU implementation, A is the only choice outside that list; the module does not separately define microcontrollers.

Why each choice is right or wrong

A - Correct: Microcontroller is not one of the three CPU components listed on p.57.

B - Incorrect: ALU performs the micro-operations (p.57).

C - Incorrect: The register set stores intermediate data (p.57).

D - Incorrect: Control unit supervises transfers and ALU operations (p.57).

Primary note proof

Study-note section COA01

> The CPU contains an ALU, control unit and registers; a microcontroller is a complete integrated control system, not another mandatory CPU subunit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Addis Ababa University mock exam · Question 40

Question 11 | Q00139

After CPU setup, which transfer method moves a block between an I/O device and memory without the CPU copying each data word?

A. Software interrupts

B. Interrupt-driven I/O

C. Polled I/O

D. Direct memory access (DMA)

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Direct memory access (DMA)

DMA assigns the block movement to a controller after the CPU configures the transfer. Polling and ordinary interrupt-driven I/O still make the CPU execute item transfers. A software interrupt is an instruction-requested control transfer to a handler, not a block-copy engine. CPU setup and completion handling remain part of DMA; bypass refers specifically to avoiding CPU copying for every word.

Why each choice is right or wrong

A - Incorrect: A software interrupt requests handler execution; it does not itself move a block directly between device and memory.

B - Incorrect: The interrupt handler still uses CPU execution to service the device and move items.

C - Incorrect: The CPU checks status and moves items itself in polled I/O.

D - Correct: The DMA controller performs the block transfer after setup, avoiding per-word CPU copies.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA19

> A software interrupt is deliberately requested by an instruction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on

the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
• Whether to read or write
• Device address
• Starting address of memory block for data
• Amount of data to be transferred The CPU carries on with other work.
• Thus DMA controller steals the CPU‟s work of I/O operation.
• The DMA module transfers the entire block of data
• One word at a time, directly to or from memory, without going through CPU.
When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically

- I/O module does not inform CPU directly
- I/O module does not interrupt CPU

Question sources

Addis Ababa University mock exam · Question 41

Question 12 | Q00140

In the simple CPU-master bus model, which bus carries instruction or operand values in both directions during memory reads and writes?

A. Address bus

B. Control bus

C. Data bus

D. None of the above

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Data bus

Data signals carry the values transferred between processor and memory. A read brings those values toward the CPU, while a write sends them toward memory, so the data bus serves both directions. Address signals select a location. Control signals coordinate operations and can include signals in both directions, but the question specifically asks about instruction or operand values rather than aggregate control direction.

Why each choice is right or wrong

A - Incorrect: The address bus selects locations rather than carrying payload values.

B - Incorrect: Control signals coordinate transfers rather than carrying the instruction or operand payload.

C - Correct: The data bus carries values toward the CPU on reads and toward memory on writes.

D - Incorrect: The data bus supplies the requested role.

Primary note proof

Study-note section COA12

> Address signals select a location; data signals carry values; control signals coordinate read/write, timing, requests and acknowledgements.

Study-note section COA12

> In the simple CPU-master model, addresses flow from CPU to memory, while data travels in either direction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 105 | lines 2-6

> • A read command: it causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer. The CPU then gets the data items by requesting I/O module to place it on the data bus.
> • A write command: it causes the I/O module to take an item of data from the data bus and subsequently transmit the data item to the peripheral.

Computer Organization and Architecture module | PDF page 102 | lines 20-24

> I/O module must have the capability to engage in communication with the CPU and external device. Thus CPU communication involves
> Command decoding: The I/O module accepts commands from the CPU carried on the control bus.
> Data: data are exchanged between the CPU and the I/O module over data bus

Computer Organization and Architecture module | PDF page 103 | lines 2-9

> Status reporting:. I/O module can report with the status signals.common used status signals are BUSY or READY. Various other status signals may be used to report various error conditions.
> Address recognition:
> just as each memory word has an address, there is address associated with every I/O device. Thus I/O module must be recognized with a unique address for each peripheral it controls. The I/O module must also be able to perform device communication. This communication involves commands,status information, and data. Some of the essentials tasks are listed

Computer Organization and Architecture module | PDF page 82 | lines 13-17

> 6.2.1 Connection of RAM and CPU
> Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Question sources

Addis Ababa University mock exam · Question 42

Question 13 | Q00365

Which specific instruction-processing technique overlaps fetch or decode of a later instruction with execution of an earlier instruction?

A. Superscalar processing

B. None of the above

C. Parallel Computation

D. Pipe-lining

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Pipe-lining

Fetching or decoding one instruction while another is being executed overlaps different instruction stages, which is pipelining. Superscalar processing instead permits issuing multiple instructions to execution resources during a cycle. These techniques can coexist, but the specific fetch/decode-versus-execute overlap identifies the pipeline mechanism rather than the processor's issue width.

Why each choice is right or wrong

A - Incorrect: Superscalar processing names multiple-issue execution resources; the stated overlap of different stages specifically describes pipelining.

B - Incorrect: Pipelining is an available matching choice.

C - Incorrect: Parallel computation is broader simultaneous processing (p.114); it does not identify the staged instruction mechanism as precisely as D.

D - Correct: The instruction pipeline reads consecutive instructions while previous ones execute (p.117).

Primary note proof

Study-note section COA35

> "Fetch/decode one instruction while executing another" describes pipelining, whereas issuing several instructions in one cycle requires multiple-issue resources.

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Question sources

Computer Science model exam · Question 96

University of Gondar model exam · Question 96

Question 14 | Q00366

Which instruction class, where supported, bypasses the immediately following instruction when its condition is met?

A. Branch instruction

B. Return instruction

C. Skip instruction

D. Jump instruction

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Skip instruction

A SKIP instruction conditionally bypasses the next instruction. That differs from an ordinary branch, which chooses a target, and from RETURN, which restores a saved return point. The original phrase jump to the next instruction did not express bypassing it; the revised wording states that intended control-flow effect precisely. The classification is based on behavior rather than an unstated machine-specific opcode.

Why each choice is right or wrong

A - Incorrect: A branch chooses a target rather than naming the specific next-instruction-bypass class.

B - Incorrect: RETURN restores a saved return point.

C - Correct: SKIP is the conditional next-instruction-bypass operation described.

D - Incorrect: A general jump chooses a target rather than naming this specific conditional operation.

Primary note proof

Study-note section COA19

> Branch instructions choose a target, conditionally or unconditionally.

Study-note section COA19

> CALL saves a return point; RETURN restores it.

Study-note section COA19

> SKIP, where supported, bypasses the next instruction when its condition holds.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 55 | lines 2-8

> 1. Fetch an instruction from memory
> 2. Decode the instruction
> 3. Read the effective address from memory if the instruction has an indirect address
> 4. Execute the instruction
> Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3.

Execute the instruction
There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 72 | lines 26-32

Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Computer Organization and Architecture module | PDF page 52 | lines 2-6

Table 5-2: Basic computer instructions
Arithmetic, logical and shift instructions provide computational capabilities for processing the type of data that the user may wish to employ. The bulk of the binary information in a digital computer is stored in memory, but all computations are done in processor registers. Therefore, the user must have the capability of moving information between these two units

Question sources

Computer Science model exam · Question 97

University of Gondar model exam · Question 97

Question 15 | Q00367

An interrupt that is caused by register overflow, attempt to divide by zero, an invalid operation code, stack overflow can be?

A. None of the above

B. Software interrupt

C. External interrupt

D. Internal interrupt

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Internal interrupt (synchronous exception).

COA19 classifies divide errors and illegal opcodes as synchronous internal exceptions because they arise from the executing instruction. Register or stack overflow belongs to that category when the machine traps it. Overflow does not invariably interrupt: ordinary flags may merely record it. Deliberate software requests and asynchronous external device events describe different causes.

Why each choice is right or wrong

A - Incorrect: The internal-exception category directly covers these instruction-related faults when trapping is enabled.

B - Incorrect: A software interrupt is deliberately requested by an instruction, rather than caused by an execution fault.

C - Incorrect: External interrupts originate from devices or timers, rather than the faulting instruction.

D - Correct: These execution-generated faults are internal exceptions; overflow trapping still depends on the architecture.

Primary note proof

Study-note section COA19

> External interrupts arise from devices or timers; synchronous internal exceptions arise from the executing instruction, such as divide errors, illegal opcodes or access violations.

Study-note section COA19

> A software interrupt is deliberately requested by an instruction.

Study-note section COA19

> Arithmetic overflow trapping depends on architecture and instruction; ordinary overflow flags need not trigger an interrupt.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

> with the higher priority first.
> Interrupt Driven I/O.
> Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for

coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 107 | lines 10-17

7.10 Direct Memory Access (DMA)
Interrupt driven and programmed I/O require active CPU intervention.
Transfer rate is limited
CPU is tied up DMA is the solution for these problems
Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Question sources

Computer Science model exam · Question 98

University of Gondar model exam · Question 98

Question 16 | Q00368

Computer address bus is

A. Unidirectional

B. None of the above

C. Multidirectional

D. Bidirectional

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Unidirectional

For the CPU-initiated memory operations described, CPU-held address registers supply the memory address (PDF pp.82-83 and 70). Thus the address path runs CPU→memory; this is a direction within that model, not a rule excluding DMA bus masters.

Why each choice is right or wrong

A - Correct: CPU loads MAR to select memory; the addressed memory does not return an address (pp.82-83).

B - Incorrect: The described CPU-address path supplies a matching option A.

C - Incorrect: The module's CPU-memory operation does not require simultaneous multidirectional address flow.

D - Incorrect: Read/write data can go both ways, but the initiating CPU supplies the address.

Primary note proof

Study-note section COA12

> In the simple CPU-master model, addresses flow from CPU to memory, while data travels in either direction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

> 6.2.1 Connection of RAM and CPU
> Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 83 | lines 11-14

> The time that elapses between the initiation of an operation and completion of that operation is called memory access time. The minimum time delay between two successive memory operations is called memory cycle time. The cycle time is usually slightly lower than the access time.

Computer Organization and Architecture module | PDF page 83 | lines 2-7

> During a “memory cycle“ n bits of data are transferred between main memory and CPU. This transfer takes place over the processor bus, which has k address lines and n data lines. The CPU initiates a memory operation by loading the appropriate data into registers MDR and MAR, and setting either Read or Write memory control line to 1. When the required operation is completed the memory control circuitry sends Memory Function Completed (MFC) signal to CPU.

Question sources

Computer Science model exam · Question 99

University of Gondar model exam · Question 99

Question 17 | Q00369

The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part specifies the operation to be performed and the second specifies .

A. address

B. output

C. Input

D. operand

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: address

The module reproduces this two-part instruction description: operation first, address second; that address locates the operand (PDF p.47). The operand itself and its address must not be conflated.

Why each choice is right or wrong

A - Correct: The second field specifies an address (p.47).

B - Incorrect: Output is not the second field in this format.

C - Incorrect: Input is not the second field in this format.

D - Incorrect: The operand is fetched using the address; it is not generally embedded in this field (p.47).

Primary note proof

Study-note section COA-S03

> In the textbook single-accumulator organization, a memory-reference instruction has an operation part and an address part.

Study-note section COA-S03

> The address locates a memory operand; the operation combines that operand with the accumulator as required.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

> The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one

section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Question sources

Computer Science model exam · Question 100

University of Gondar model exam · Question 100

Question 18 | Q00370

Which of the following computer memory is fastest?

A. RAM

B. Register

C. None of the above

D. Hard disk

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Register

The hierarchy figure on PDF p.79 places registers before cache, RAM and disk. Register organization also avoids time-consuming memory accesses (p.58). Although p.84 calls cache the fastest memory-hierarchy component, it qualifies cache as approaching CPU component speed; registers are the CPU storage meant here.

Why each choice is right or wrong

A - Incorrect: RAM requires memory access, avoided by retaining intermediate values in registers (p.58).

B - Correct: Registers are the fastest offered CPU storage tier in Figure 6.1 (p.79).

C - Incorrect: A listed choice fits.

D - Incorrect: Disk lies at the slow, large secondary-storage end of the hierarchy (p.79).

Primary note proof

Study-note section COA23

A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

6.1 Memory Hierarchy:
A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 58 | lines 4-11

In the programming examples of the previous chapter, we have shown that memory locations are needed for storing pointers, counters, return addresses, temporary results and partial product during multiplication. Having to refer to memory locations for such applications is time consuming because memory access is the most time consuming operation in a computer. It is more convenient and more efficient to store these intermediate values in processor register. When a large number of registers are included in the CPU, It is most efficient to connect them through a common bus system. The registers communicate with each other not only for direct data transfers, but also while performing various micro operations.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science model exam · Question 101

University of Gondar model exam · Question 101

Question 19 | Q00539

It is a method used to synchronize the I/O device with the microprocessor

A. Interrupt

B. Handshaking

C. Interface

D. None

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Handshaking.

COA31 describes asynchronous handshaking as an exchange of request/ready and acknowledgement signals. The sender presents data, the receiver accepts it, and the feedback coordinates transfer despite unequal device and processor speeds. An interrupt requests service but does not itself name this two-sided transfer protocol. An interface is the broader connection exposing the signals and registers.

Why each choice is right or wrong

A - Incorrect: Interrupts request processor attention; they do not specifically identify the request-and-acknowledgement transfer protocol.

B - Correct: Request and acknowledgement signals synchronize the sender and receiver despite unequal operating speeds.

C - Incorrect: An interface provides the connection; handshaking names the synchronization method implemented through it.

D - Incorrect: Handshaking is an explicitly supported synchronization method, so an affirmative option fits.

Primary note proof

Study-note section COA31

> Asynchronous handshaking uses request/ready and acknowledgement signals to tolerate unequal speeds: sender presents data and requests, receiver accepts and acknowledges, then each withdraws its signal.

Study-note section COA31

> A single strobe lacks this feedback.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 101 | lines 2-9

> Asynchronous data transfer
> The internal operation in digital system is synchronized by means of clock pulse supplied by a common pulse generator. Two units such as cpu and an I/O are designed independently of each other, if the register in the interface share a common clock with the cpu registers, the transfer between the two units is called to be synchronous.in most case the internal timing in each unit is independent from the other in that each uses its own private clock for internal register.in such cases the two units are said to be asynchronous to each other. This approach is widely used in most computers systems.

Computer Organization and Architecture module | PDF page 101 | lines 10-16

> 7.4 Input/output Problems
> Wide variety of peripherals and delivering different amounts of data per second Work at different speeds Send/receive data in different formats all slower than CPU and RAM. Hence I/O modules are used as a solution.
> Input/output Module: It is the entity within a computer that is responsible for the control of one or more external devices and for the exchange of data between those devices and main memory and/or CPU.

Question sources

Computer Science model exam · 2015 · Question 8

Question 20 | Q00542

Which one of the following instructions is incorrect?

A. MOV AL, BP

B. INC CX

C. DEC CX

D. All

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: MOV AL, BP is invalid.

Under the ordinary Intel forms specified in COA22, AL is an 8-bit destination while BP is a 16-bit source. MOV does not implicitly resize these operands, so MOV AL, BP is invalid. INC CX and DEC CX are legal 16-bit operations. Consequently the all-invalid alternative fails because two listed instructions are valid.

Why each choice is right or wrong

A - Correct: The 8-bit AL and 16-bit BP widths do not match for this MOV instruction.

B - Incorrect: INC CX legally increments a 16-bit register under the stated legacy x86 assumptions.

C - Incorrect: DEC CX legally decrements a 16-bit register under the stated legacy x86 assumptions.

D - Incorrect: INC CX and DEC CX are valid, so the instructions are not all incorrect.

Primary note proof

Study-note section COA22

> MOV AL,BP is invalid because operand widths differ (8 versus 16 bits).

Study-note section COA22

> INC CX and DEC CX are valid word operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

> Data transfer instruction.
> Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Computer Organization and Architecture module | PDF page 72 | lines 2-19

> Immediate Mode: In this mode the operand is specified in the instruction itself. In other words, an immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction. Immediate mode instructions are useful for initializing registers to a constant value.
> When the address field specifies a processor register, the instruction is said to be in the register mode.
> Register Mode: In this mode the operands are in registers that reside within the CPU. The particular register is selected from a register field in the instruction. A k-bit field can specify any of the 2k registers.
> Register Indirect Mode: In this mode the instruction specifies a register in the CPU whose contents give the address of the operand in memory. In other words, the selected register contains the address of the operand rather than the operand itself. Before using a register indirect mode instruction, the programmer must ensure that the memory address of the operand is placed in the processor register with a previous instruction. A reference to the register is then equivalent to specifying a memory address. The advantage of the register indirect mode instruction is that the address field of the instruction uses fewer bits to select a register than would have been required to specify a memory address directly.

Computer Organization and Architecture module | PDF page 38 | lines 8-12

Digital modules are best defined by the registers they contain and the operations that are performed on the data stored in them. The operations executed on data stored in registers are called micro operations. A micro operation is an elementary operation performed on the information stored in one or more registers.
Examples of micro operation are shift, count, clear, and load.

Computer Organization and Architecture module | PDF page 43 | lines 3-12

R3←R1 + R2 Contents of R1 plus R2 transferred to R3
R3←R1 -R2 Contents of R1 minus R2 transferred to R3
R2←$\bar{}$Complement the contents of R2 (1's complement)
R2←$\bar{}$+ 1 2's complement the contents of R2 (negative)
R3←R1 +$\bar{}$+ 1 R1 plus the 2's complement of R2 (substruction)
R1←R1 + 1 Increment the contents of R1 by one
R1←R1- 1 Decrement the contents of R1 by one
Table 4.2: Arithmetic Micro operation
The increment and decrement micro operations are symbolized by plus one and minus- one operation, respectively.

Question sources

Computer Science model exam · 2015 · Question 11

Question 21 | Q00544

A register used to indicate the condition of the microprocessor is

A. SI

B. FLAG

C. DI

D. EIP

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: FLAG register.

COA21 states that FLAGS/EFLAGS/RFLAGS report processor status. This is the register family indicated by the option FLAG. SI and DI are general-purpose registers with conventional index roles, and EIP identifies the instruction position. Neither the index registers nor the instruction pointer is the processor condition/status register requested.

Why each choice is right or wrong

A - Incorrect: SI has an index/general-purpose role rather than collecting the processor condition flags.

B - Correct: FLAG denotes the status-register family reporting processor conditions such as arithmetic status.

C - Incorrect: DI has an index/general-purpose role rather than serving as the processor status register.

D - Incorrect: EIP identifies instruction position; it does not contain the collection of processor condition flags.

Primary note proof

Study-note section COA21

> FLAGS/EFLAGS/RFLAGS report status; IP/EIP/RIP identifies instruction position.

Study-note section COA21

> In Intel x86 notation, AX/BX/CX/DX, BP/SP and SI/DI are 16-bit general-purpose registers, with conventional special roles.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

> Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction

There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

Computer Science model exam · 2015 · Question 13

Question 22 | Q00545

Which one of the following instructions is correct

A. MOV DS, 20

B. MOV ES, 40

C. MOV SS, 60

D. None

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: None of these immediate-to-segment MOV forms is valid.

COA22 excludes a direct immediate-to-segment-register MOV in the assumed legacy x86 syntax. DS, ES and SS are segment registers, and each listed source is an immediate number. A permitted sequence first loads the value into a suitable general register and then moves that register to the segment register. Therefore all three displayed forms fail.

Why each choice is right or wrong

A - Incorrect: MOV DS, 20 attempts the unsupported direct transfer of an immediate into a segment register.

B - Incorrect: MOV ES, 40 also attempts an unsupported immediate-to-segment-register transfer in the stated syntax.

C - Incorrect: MOV SS, 60 has the same unsupported direct immediate source for a segment destination.

D - Correct: All three proposed segment-register instructions require an intermediate register or supported memory source.

Primary note proof

Study-note section COA22

Segment-register MOV has register/memory forms, not a direct immediate-to-segment form: load an immediate into AX, then MOV DS,AX.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 72 | lines 2-19

> Immediate Mode: In this mode the operand is specified in the instruction itself. In other words, an immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction. Immediate mode instructions are useful for initializing registers to a constant value.
> When the address field specifies a processor register, the instruction is said to be in the register mode.
> Register Mode: In this mode the operands are in registers that reside within the CPU. The particular register is selected from a register field in the instruction. A k-bit field can specify any of the 2k registers.
> Register Indirect Mode: In this mode the instruction specifies a register in the CPU whose contents give the address of the operand in memory. In other words, the selected register contains the address of the operand rather than the operand itself. Before using a register indirect mode instruction, the programmer must ensure that the memory address of the operand is placed in the processor register with a previous instruction. A reference to the register is then equivalent to specifying a memory address. The advantage of the register indirect mode instruction is that the address field of the instruction uses fewer bits to select a register than would have been required to specify a memory address directly.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

> Data transfer instruction.
> Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Question sources

Computer Science model exam · 2015 · Question 14

Question 23 | Q00546

A register used to address the next instruction in a section of memory is

A. RSP

B. CS

C. RIP

D. SS

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: RIP, the instruction pointer.

COA21 identifies IP/EIP/RIP as the instruction-position register family; COA14 identifies the PC role as locating the next instruction to fetch. RIP is the corresponding 64-bit x86 instruction pointer. CS selects the code segment and SS the stack segment, while RSP is the stack pointer. A segment selector alone does not identify the next instruction offset.

Why each choice is right or wrong

A - Incorrect: RSP tracks the stack, not the next instruction position within the instruction stream.

B - Incorrect: CS identifies the code segment; the instruction pointer supplies the instruction position within it.

C - Correct: RIP is the x86 instruction pointer used to identify the next instruction position.

D - Incorrect: SS identifies the stack segment rather than the location of the next instruction.

Primary note proof

Study-note section COA-S05

> In Intel 64-bit mode, RIP holds the offset of the next instruction to execute.

Study-note section COA-S05

> RSP is the 64-bit stack pointer, extending the ESP/SP stack role.

Study-note section COA-S05

> These pointers are distinct from CS and SS, which are code and stack segment registers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 69 | lines 15-23

> 5.15.4 Zero-Address Instruction
> A stack-organized computer does not use an address field for instructions ADD and MUL. The PUSH and POP instructions, however, need an address field to specify the operand that

communicates with the stack. The following program shows how X = (A+B) * (C+D) will be written for a stack-organized computer (TOS stands for top-of-stack)
PUSH A TOS← A, PUSH B TOS ←B, ADD TOS ←(A + B),
PUSH C TOS ←C
PUSH D TOS ←D, ADD TOS ←(C + D) , MUL TOS ←(C + D) (A + B),
POP X M[X] ←TOS.

Question sources

Computer Science model exam · 2015 · Question 15

Question 24 | Q00562

The Intel's first true 32-bit microprocessor is

A. Pentium

B. 80486

C. 80286

D. 80386

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer D: 80386, within the offered x86 lineage.

COA38 explicitly dates 32-bit x86 execution to the 80386 in 1985. It precedes the 80486 and Pentium, while the 80286 is a 16-bit predecessor. The offered choices therefore select 80386. This conclusion concerns the x86 sequence represented by the options; it should not be expanded into a claim about every historical Intel processor family.

Why each choice is right or wrong

A - Incorrect: Pentium follows the 80386; it was not the first 32-bit member among these x86 choices.

B - Incorrect: The 80486 follows the 80386, which had already introduced 32-bit x86 execution.

C - Incorrect: The note identifies the 80286 as a 16-bit predecessor, rather than the 32-bit introduction.

D - Correct: The 80386 introduced 32-bit x86 execution before the other offered 32-bit processors.

Primary note proof

Study-note section COA38

Intel’s 80386 introduced 32-bit x86 execution in 1985, before 80486/Pentium; the 80286 was a 16-bit predecessor.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Computer Science model exam · 2015 · Question 31

Question 25 | Q00569

The content of BH is 10, AL is 8, CL is 4 how much is the content of AX after MUL BH

A. 40

B. 80

C. 32

D. 320

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: AX = 80.

The operand BH is a byte register. COA22 states that unsigned byte MUL multiplies AL by the explicit byte operand and stores the product in AX. Substituting the given values yields 8 × 10 = 80. CL is not an operand of MUL BH; multiplying by its value would change the instruction being evaluated.

Why each choice is right or wrong

A - Incorrect: 40 multiplies BH by CL, but byte MUL uses AL as its implicit multiplicand.

B - Correct: AL × BH equals 8 × 10 = 80, written into AX.

C - Incorrect: 32 multiplies AL by CL, whereas the explicit operand is BH, not CL.

D - Incorrect: 320 incorrectly includes CL as a third factor; MUL BH multiplies only AL and BH.

Primary note proof

Study-note section COA22

> Unsigned MUL with a byte operand multiplies AL by that byte and writes AX.

Study-note section COA22

> Thus AL=8 and BH=10 gives AX=80 after MUL BH; CL is irrelevant.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 74 | lines 3-14

> Data Manipulation
> Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
> 1. Arithmetic instruction
> 2. Logical and bit manipulation
> 3. Shift instruction
> a. Arithmetic Instruction
> The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

> Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Question sources

Computer Science model exam · 2015 · Question 38

Question 26 | Q00573

Which legacy x86 instruction is arithmetic rather than a data-transfer or operand-exchange operation?

A. MOVS

B. LODS

C. XCHG

D. INC

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: INC

MOVS and LODS are string-transfer operations, and XCHG exchanges operand values. INC instead increments an operand, making it arithmetic. The revised stem states the classification criterion, so different no longer depends on whether the reader groups string operations, operand counts or another property. This retains the original instruction set and intended comparison while giving every choice a definite classification.

Why each choice is right or wrong

A - Incorrect: MOVS transfers string data.

B - Incorrect: LODS loads string data.

C - Incorrect: XCHG exchanges operands rather than performing arithmetic.

D - Correct: INC increments an operand arithmetically.

Primary note proof

Study-note section COA22

> MOVS and LODS are string transfer operations, XCHG exchanges operands, and INC is arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

> Data transfer instruction.
> Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Question sources

Computer Science model exam · 2015 · Question 42

Question 27 | Q00609

Which register is a user-accessible register?

A. DX

B. IR

C. MBR

D. MDR

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: DX is programmer-accessible.

COA21 explicitly identifies DX as programmer-visible and IR/MDR as conceptual internal registers. MBR is the memory-buffer name used for the same data-transfer role as MDR in COA11. The question asks which register is user-accessible in this architectural sense, so the x86 general-purpose register DX is the supported choice.

Why each choice is right or wrong

A - Correct: DX belongs to the programmer-visible x86 register set and is usable by ordinary instructions.

B - Incorrect: IR is the conceptual internal instruction register, rather than an ordinary programmer-addressable register.

C - Incorrect: MBR is the internal memory-buffer/data-transfer register, not a user-visible general-purpose register.

D - Incorrect: MDR is the conceptual internal memory-data register, not an ordinary user-accessible execution register.

Primary note proof

Study-note section COA21

DX is programmer-visible; IR/MDR are conceptual internal registers.

Study-note section COA11

A read places an address in MAR/AR, asserts read, waits for memory and captures data in MDR/MBR/DR.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

6.2.1 Connection of RAM and CPU
Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Question sources

Computer Science model exam · 2015 · Question 78

Question 28 | Q00614

Which register is found starting from Pentium processor?

A. EAX

B. EIP

C. R8

D. None

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: None was introduced with original Pentium.

Reading starting from as introduced with Pentium, COA21 rules out every named register. EAX and EIP already existed with the 80386, while R8 belongs to the later 64-bit extension rather than original Pentium. Therefore none is correct under the introduction-date reading. Simply asking which registers a Pentium contains would be a different question, with EAX and EIP both present.

Why each choice is right or wrong

A - Incorrect: EAX existed before Pentium in the 80386; it was not introduced starting with Pentium.

B - Incorrect: EIP also existed in the 80386, so Pentium was not its first generation.

C - Incorrect: R8 belongs to the later 64-bit extension, not the original Pentium register set.

D - Correct: None of the named registers was introduced with original Pentium under the starting-generation reading.

Primary note proof

Study-note section COA21

> EAX/EIP existed with the 80386, before Pentium; R8-R15 belong to the 64-bit extension, not original Pentium.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Computer Science model exam · 2015 · Question 83

Question 29 | Q00621

How many physical processor cores does the Intel Core i7-920 desktop processor have? Count cores, not hardware threads.

A. 5

B. 4

C. 6

D. 7

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: 4

The supplementary Intel desktop-processor comparison chart specifies four cores and eight threads for the i7-920. The answer is therefore four physical cores. The supplied note provides related background that Core i7 is a family with no single core count, but does not supply this model's number. Naming the model and consulting Intel supplies that missing fact without implying that every Core i7 has the same configuration.

Why each choice is right or wrong

A - Incorrect: Five does not match Intel's physical-core count for this model.

B - Correct: Intel lists four physical cores for the i7-920.

C - Incorrect: Six is not the specified model's physical-core count.

D - Incorrect: Seven is not this model's physical-core count.

Primary note proof

Study-note section COA-S01

> Intel's desktop comparison chart lists the Core i7-920 with four physical cores and eight hardware threads.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 114 | lines 2-10

> The following are important points about parallel processing:
> • A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
> • The system may have two or more ALUs and be able to execute two or more instructions at the same time
> • Also, the system may have two or more processors operating concurrently
> • Goal is to increase the throughput - the amount of processing that can be

accomplished during a given interval of time
Parallel processing can be classified from:

Question sources

Computer Science model exam · 2015 · Question 90

Question 30 | Q00628

A memory operand has effective address EA=A, where A is the address field stored directly in the instruction. Which addressing mode is used?

A. Direct addressing

B. Immediate addressing

C. Register addressing

D. None

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Direct addressing

In direct addressing, the instruction's address field is itself the effective address: EA=A, and the operand is M[A]. Immediate addressing uses A as a value rather than a memory address, while register addressing uses a register as the operand source. A transfer's register and memory endpoints alone do not determine its addressing mode, so the revised question supplies the missing effective-address relationship.

Why each choice is right or wrong

A - Correct: EA=A defines direct addressing.

B - Incorrect: Immediate addressing uses A as the operand value.

C - Incorrect: Register addressing uses a register operand rather than M[A].

D - Incorrect: Direct addressing matches the relationship.

Primary note proof

Study-note section COA18

Immediate uses value A; direct uses EA=A and operand M[A]; indirect uses EA=M[A] then operand M[EA].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 72 | lines 26-32

Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Computer Organization and Architecture module | PDF page 72 | lines 2-19

Immediate Mode: In this mode the operand is specified in the instruction itself. In other words, an immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction. Immediate mode instructions are useful for initializing registers to a constant value.
When the address field specifies a processor register, the instruction is said to be in the register mode.
Register Mode: In this mode the operands are in registers that reside within the CPU. The particular register is selected from a register field in the instruction. A k-bit field can specify any of the 2k registers.
Register Indirect Mode: In this mode the instruction specifies a register in the CPU whose contents give the address of the operand in memory. In other words, the selected register contains the address of the operand rather than the operand itself. Before using a register indirect mode instruction, the programmer must ensure that the memory address of the operand is placed in the processor register with a previous instruction. A reference to the register is then equivalent to specifying a memory address. The advantage of the register indirect mode instruction is that the address field of the instruction uses fewer bits to select a register than would have been required to specify a memory address directly.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

Data transfer instruction.
Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Question sources

Computer Science model exam · 2015 · Question 97

Question 31 | Q00630

One of the following is not a characteristic of SRAM

A. Fast

B. Expensive

C. Memory refreshing s

D. None

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Memory refreshing s

SRAM stores bits in flip-flop state; applying power is sufficient and no refresh is needed (PDF p.81). Fast and relatively expensive are expressly listed, making refreshing the exception.

Why each choice is right or wrong

A - Incorrect: Faster than DRAM is stated (p.81).

B - Incorrect: More expensive than DRAM is stated (p.81).

C - Correct: No need for refreshing is explicit (p.81), so this is the non-characteristic.

D - Incorrect: There is a non-characteristic, C.

Primary note proof

Study-note section COA24

> SRAM uses powered bistable storage: fast and costly per bit, without periodic refresh.

Study-note section COA24

> DRAM stores charge and requires refresh: denser and commonly used for main memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 81 | lines 6-16

> Static RAM:
> • a bit of data is stored using the state of a flip-flop.
> • Applying power is enough (no need for refreshing).
> • Retains value indefinitely, as long as it is kept powered.
> • Mostly uses to create cache memory of CPU
> • Faster and more expensive than DRAM.
> ROM
> ROM is used for storing programs that are permanently resident in the computer and for tables of constants that do not change in value once the production of the computer is

completed The ROM portion of main memory is needed for storing an initial program called bootstrap loader, which is to start the computer software operating when power is

Computer Organization and Architecture module | PDF page 80 | lines 5-11

DYNAMIC RAM (DRAM)
is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Question sources

Computer Science model exam · 2015 · Question 99

Question 32 | Q00632

Which one of the following is analog to digital converter?

A. ADC080X

B. DAC0830

C. 82C55

D. None

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer A: ADC080X is the analog-to-digital converter.

COA33 defines an ADC as producing a digital code from an analog input and specifically identifies ADC080x as an 8-bit successive-approximation ADC family. DAC0830 performs the reverse conversion. The 82C55 is a programmable parallel I/O interface, not an ADC. The component identifiers therefore resolve this question without inferring capabilities merely from their general interface role.

Why each choice is right or wrong

A - Correct: ADC080x is explicitly identified as the analog-to-digital converter family in the note.

B - Incorrect: DAC0830 converts digital codes into analog output, which is the reverse direction.

C - Incorrect: 82C55 provides programmable parallel I/O; the note explicitly distinguishes it from an ADC.

D - Incorrect: ADC080X is a valid analog-to-digital component, so the list is not missing an answer.

Primary note proof

Study-note section COA33

An ADC measures an analog input and produces a digital code; a DAC produces an analog output from a digital code.

Study-note section COA33

ADC080x parts are 8-bit successive-approximation analog-to-digital converters; DAC0830 is a digital-to-analog converter, while 82C55 denotes a programmable parallel I/O interface, not an ADC.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Question sources

Computer Science model exam · 2015 · Question 101

Question 33 | Q00648

Which is NOT an Intel x86 16-bit general-purpose register?

A. AX

B. BP

C. SI

D. FLAGS

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: FLAGS

Intel’s general-purpose classification includes AX, BP and SI even though these registers have conventional special roles. FLAGS instead reports status conditions and is outside that group. The final option supplies an actual non-general-purpose register rather than incorrectly excluding SP merely because it tracks the stack. This preserves the register-class distinction without inventing a narrower meaning for multipurpose or misrepresenting Intel’s classification.

Why each choice is right or wrong

A - Incorrect: AX is a 16-bit general-purpose register.

B - Incorrect: BP is general-purpose despite its conventional base-pointer role.

C - Incorrect: SI is general-purpose despite its conventional source-index role.

D - Correct: FLAGS reports processor status rather than belonging to the general-purpose group.

Primary note proof

Study-note section COA21

> In Intel x86 notation, AX/BX/CX/DX, BP/SP and SI/DI are 16-bit general-purpose registers, with conventional special roles.

Study-note section COA21

> FLAGS/EFLAGS/RFLAGS report status; IP/EIP/RIP identifies instruction position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 69 | lines 15-23

> 5.15.4 Zero-Address Instruction
> A stack-organized computer does not use an address field for instructions ADD and MUL. The PUSH and POP instructions, however, need an address field to specify the operand that communicates with the stack. The following program shows how X = (A+B) * (C+D) will be written for a stack-organized computer (TOS stands for top-of-stack)

PUSH A TOS← A, PUSH B TOS ←B, ADD TOS ←(A + B),
PUSH C TOS ←C
PUSH D TOS ←D, ADD TOS ←(C + D) , MUL TOS ←(C + D) (A + B),
POP X M[X] ←TOS.

Question sources

Computer Science model exam · 2015 · Question 117

Question 34 | Q00689

Which of the following is a type of computer architecture?

A. Microarchitecture

B. Harvard Architecture

C. Von-Neumann Architecture

D. All of the mentioned

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: All listed categories are forms of computer design/architecture in the broad wording.

COA01 calls microarchitecture the internal processor design, while COA02 contrasts Harvard and Von Neumann memory organizations. The broad phrase type of computer architecture encompasses all three usages, although they describe different abstraction levels. The comprehensive answer is therefore all of the mentioned. No single named option covers the entire set of valid architectural categories.

Why each choice is right or wrong

A - Incorrect: Microarchitecture is a valid internal-design category, but this alone omits the two valid memory organizations.

B - Incorrect: Harvard is a valid architecture category, but selecting it alone excludes the other valid categories.

C - Incorrect: Von Neumann is valid, but the broad question also admits microarchitecture and Harvard organization.

D - Correct: All three names describe architectural design categories, though they operate at different abstraction levels.

Primary note proof

Study-note section COA01

Microarchitecture names the internal processor design.

Study-note section COA02

In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA02

Harvard separates instruction and data memories or paths.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 5 | lines 12-20

Computer architecture: refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory.
Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

Question sources

Computer Science model exam 2 · Question 30

Question 35 | Q00690

Which one is not a combinational circuit?

A. Encoder

B. Decoder

C. Multiplexer

D. De-multiplexer

E. Register

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer E: Register

A register is a group of state-holding flip-flops (PDF p.35). Sequential circuits retain past information (p.15), whereas code/routing combinational circuits transform current inputs (pp.31,33-34). Thus register is the non-combinational choice.

Why each choice is right or wrong

A - Incorrect: Encoder maps the active input into a binary code (p.34); no stored state is part of this described mapping.

B - Incorrect: Decoder selects an output based on the input pattern (p.33).

C - Incorrect: Multiplexer is explicitly combinational (p.31).

D - Incorrect: Demultiplexer routes one input to a selected output (p.33), a present-input routing function.

E - Correct: Register contains storage flip-flops and retains state (p.35).

Primary note proof

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Study-note section COA05

> A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

> • In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
> • However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
> • Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
> • Such circuits include storage elements that store the values of logic signals.
> • Flip-flops can store a one-bit data (information).

Computer Organization and Architecture module | PDF page 35 | lines 3-11

> 3.9 Registers
> • A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
> • A register is a digital circuit used within the CPU to store one or more bits of data.
> • An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
> • In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
> • Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 33 | lines 7-14

> 3.7 Decoders
> • A decoder is a combinational circuit with a number of output lines, only one of which

is selected at any time, depending on the pattern of input lines.
• In general, a decoder has n inputs and 2n
outputs.
• Decoders find many uses in digital computers.
• One example is address decoding.
• The other is binary-to- octal conversion.

Computer Organization and Architecture module | PDF page 34 | lines 3-8

3.8 Encoders
• An encoder is a digital circuit that performs the inverse operation of a decoder.
• An encoder has 2n
(or less) input lines and n output lines. The output lines generate
the binary code corresponding to the input value.
• An example of an encoder is the octal-to-binary encoder.

Computer Organization and Architecture module | PDF page 31 | lines 12-16

3.5 Multiplexers
• Multiplexer is a combinational circuit that receives binary information from one of 2n input data & directs to one output.
• The multiplexer connects multiple inputs to a single output.
• At any time, one of the inputs is selected to be passed to the output.

Computer Organization and Architecture module | PDF page 10 | lines 22-27

1.4 Combinational Circuits
• A combinational circuit is an interconnected set of gates whose output at any time is a function only of the input at that time. As with a single gate, the appearance of the input is followed almost immediately by the appearance of the output, with only gate delays. In general terms, a combinational circuit consists of n binary inputs and m binary outputs.

Question sources

Computer Science model exam 2 · Question 31

Question 36 | Q00691

In a _____ cache organization, cache permits data to be stored in any cache block.

A. Direct Mapped

B. Set Associativity

C. Fully Associative

D. All

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Fully Associative

Associative cache stores an address with each data item and searches by content (PDF p.87), rather than constraining a block to a single index. Set association retains index constraints (p.88). Fully associative is therefore the any-block placement organization.

Why each choice is right or wrong

A - Incorrect: Direct mapping has fixed index conflicts (p.88), so does not allow arbitrary block placement.

B - Incorrect: Set association allows multiple entries under the same index, not any block anywhere (p.88).

C - Correct: Address/content pairs searched associatively permit the unrestricted mapping described (p.87).

D - Incorrect: Direct and set mappings are restricted, so not all satisfy the claim.

Primary note proof

Study-note section COA25

> Direct mapping uses line=block mod L.

Study-note section COA25

> Fully associative placement permits any line.

Study-note section COA25

> A W-way set-associative cache has S=L/W sets and set=block mod S; any of W ways in that set may hold the block.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 87 | lines 2-15

> Associative Mapping
> • An associative mapping uses an associative memory.
> • This memory is being accessed using its contents.
> • Associative memory is expensive compared to RAM
> • Each line of cache memory will accommodate the address (main memory) and the contents of that address from the main memory.
> • That is why this memory is also called Content Addressable Memory (CAM). It allows each block of main memory to be stored in the cache.
> A CPU address of 15 bits is places in the argument register and the associative memory us searched for a matching address
> • If the address is found, the corresponding 12bits data is read and sent to the CPU

• If not, the main memory is accessed for the word
If the cache is full, an address-data pair must be displaced to make room for a pair that is needed and not presently in the cache

Computer Organization and Architecture module | PDF page 88 | lines 2-13

Set Associative Mapping
That is the easy control of the direct mapping cache and the more flexible mapping of the fully associative cache.
In set associative mapping, each cache location can have more than one pair of tag + data items.
That is more than one pair of tag and data are residing at the same location of cache memory.
If one cache location is holding two pair of tag + data items, that is called 2-way set associative mapping.
The disadvantage of direct mapping is that two words with the same index in their address but with different tag values cannot reside in cache memory at the same time
Set-Associative Mapping is an improvement over the direct-mapping in that each word of cache can store two or more word of memory under the same index address

Question sources

Computer Science model exam 2 · Question 32

Question 37 | Q00692

Which one used to translate virtual addresses into physical addresses when we are implementing the concept of Virtual memory?

A. LRU

B. FIFO

C. Memory Management Unit

D. All

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Memory Management Unit (MMU).

COA27 states that the MMU translates virtual/logical addresses into physical addresses using page tables and often a TLB. LRU and FIFO instead choose replacement victims, as described in COA26. Neither replacement policy performs address translation. Consequently the MMU is the single component matching the requested virtual-memory translation role; the all alternative incorrectly includes two different mechanisms.

Why each choice is right or wrong

A - Incorrect: LRU chooses a least-recently-used replacement victim; it does not translate virtual addresses.

B - Incorrect: FIFO selects the earliest inserted replacement victim, rather than computing physical addresses.

C - Correct: The MMU performs virtual-to-physical address translation using the translation structures described in the note.

D - Incorrect: Only the MMU translates addresses; LRU and FIFO are replacement policies.

Primary note proof

Study-note section COA27

> The MMU translates virtual/logical addresses to physical addresses, using page tables and often a TLB of recent translations.

Study-note section COA26

> FIFO evicts the earliest inserted line; LRU evicts the least recently accessed line.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 90 | lines 9-23

> Concept of Virtual Memory
> A programmer can write a program which requires more memory space than the capacity of the main memory. Such a program is executed by virtual memory technique.
> The program is stored in the secondary memory. The memory management unit (MMU) transfers the currently needed part of the program from the secondary memory to the main memory for execution.
> This to and fro movement of instructions and data (parts of a program) between the main memory and the secondary memory is called Swapping.
> Address Space And Memory Space.
> Virtual address is the address used by the programmer and the set of such addresses is called the address space or virtual memory.
> An address in main memory is called a location or physical address. The set of such locations in main memory is called the memory space or physical memory.
> CPU generated logical address consisting of a logical page number plus the location within that page (x).

Computer Organization and Architecture module | PDF page 91 | lines 2-7

> It must be mapped onto an actual (physical) main memory address by the operating system using mapper.
> If the page is present in the main memory, CPU gets the required data from the main memory.
> If the mapper detects that the requested page is not present in main memory, a page fault occurs and the page must be read from secondary storage into a page frame in main memory.

Computer Organization and Architecture module | PDF page 88 | lines 14-22

> 6.3.2 Replacement Algorithms of Cache Memory
> Replacement algorithms are used when there are no available space in a cache in which to place a data. Four of the most common cache replacement algorithms are described below:
> Least Recently Used (LRU):
> The LRU algorithm selects for replacement the item that has been least recently used by the CPU.
> First-In-First-Out (FIFO):
> The FIFO algorithm selects for replacement the item that has been in the cache from the longest time.

Question sources

Computer Science model exam 2 · Question 33

Question 38 | Q00693

In a _____ bus, flow is only one-way from the CPU to memory.

A. Address Bus

B. Data Bus

C. Control Bus

D. A & C

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Address Bus

In the CPU-master model, CPU address registers identify the memory location (PDF pp.82-83), while data supports reads and writes (p.105) and control/status includes messages from both sides (pp.102-103). This makes A the specific one-way address path.

Why each choice is right or wrong

A - Correct: The CPU supplies the memory address (pp.82-83).

B - Incorrect: Read and write data travel in opposite directions (p.105).

C - Incorrect: Status reports return from I/O modules (p.103), so the whole control/status collection is not only CPU-outward.

D - Incorrect: C is not restricted to that one direction, so the conjunction fails.

Primary note proof

Study-note section COA12

> In the simple CPU-master model, addresses flow from CPU to memory, while data travels in either direction.

Study-note section COA12

Aggregate control signals include both CPU outputs and device inputs; both data and control buses can therefore be bidirectional under that scope.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

6.2.1 Connection of RAM and CPU
Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 105 | lines 2-6

• A read command: it causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer. The CPU then gets the data items by requesting I/O module to place it on the data bus.
• A write command: it causes the I/O module to take an item of data from the data bus and subsequently transmit the data item to the peripheral.

Computer Organization and Architecture module | PDF page 103 | lines 2-9

Status reporting:. I/O module can report with the status signals.common used status signals are BUSY or READY. Various other status signals may be used to report various error conditions.
Address recognition:
just as each memory word has an address, there is address associated with every I/O device. Thus I/O module must be recognized with a unique address for each peripheral it controls. The I/O module must also be able to perform device communication. This communication involves commands,status information, and data. Some of the essentials tasks are listed

Computer Organization and Architecture module | PDF page 102 | lines 20-24

I/O module must have the capability to engage in communication with the CPU and external device. Thus CPU communication involves
Command decoding: The I/O module accepts commands from the CPU carried on the control bus.
Data: data are exchanged between the CPU and the I/O module over data bus

Computer Organization and Architecture module | PDF page 83 | lines 2-7

During a “memory cycle‟ n bits of data are transferred between main memory and CPU. This transfer takes place over the processor bus, which has k address lines and n data lines. The CPU initiates a memory operation by loading the appropriate data into registers MDR and MAR, and setting either Read or Write memory control line to 1. When the required operation is completed the memory control circuitry sends Memory Function Completed (MFC) signal to CPU.

Question sources

Computer Science model exam 2 · Question 34

Question 39 | Q00694

In memory hierarchy, faster access then_____.

A. Greater cost per bit

B. Greater capacity

C. Smaller cost per bit

D. All

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer A: Greater cost per bit

The module states the speed/cost tradeoff directly: faster access has greater cost per bit (PDF p.79). Its separate capacity bullets associate larger capacity with lower cost and slower access, rather than faster access.

Why each choice is right or wrong

A - Correct: Exact stated speed/cost relationship (p.79).

B - Incorrect: Greater capacity is associated with slower access, not the stated faster tier (p.79).

C - Incorrect: Smaller cost per bit is associated with greater capacity, opposite the faster-access bullet (p.79).

D - Incorrect: B and C do not match the faster-access relationship.

Primary note proof

Study-note section COA23

> Faster levels are generally smaller and more expensive per bit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
> Figure 6.1 memory hierarchy

- Design constraints: How much? How fast? How expensive?
- Faster access time, greater cost per bit
- Greater capacity, smaller cost per bit,
- Greater capacity, slower access time.

Question sources

Computer Science model exam 2 · Question 35

Question 40 | Q00695

Which Shift register type used to act as a delay element?

A. Serial-In Serial-Out

B. Parallel-In Serial-Out

C. Serial-In Parallel-Out

D. All

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Serial-In Serial-Out

The shift register accepts/transfers data serially and moves each bit one position per clock, emitting the rightmost bit (PDF p.36). With serial input and serial output, an n-stage register delays the stream by n shifts. This derivation identifies SISO under the usual stream-delay meaning.

Why each choice is right or wrong

A - Correct: Serial input traverses storage stages to serial output, producing the clock-count delay (p.36).

B - Incorrect: Parallel input/serial output serializes a loaded word; it does not preserve the incoming serial stream in the described way.

C - Incorrect: Serial input/parallel output exposes the stored bits together rather than the delayed serial stream.

D - Incorrect: Not all listed I/O forms implement the same serial-stream delay.

Primary note proof

Study-note section COA13

SISO shift registers delay serial streams; SIPO collects a word and PISO serializes a parallel word.

Study-note section COA05

A flip-flop stores one bit while powered.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 36 | lines 2-9

Shift Register
• A shift register accepts and/or transfers information serially.
• Data are input only to the leftmost flip-flop.
• Shift registers can be used to interface to serial I/O devices.
• With each clock pulse, data are shifted to the right, one position, and the rightmost bit is transferred out.
• In addition, they can be used within the ALU to perform logical shift and rotate functions.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

3.9 Registers
• A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
• A register is a digital circuit used within the CPU to store one or more bits of data.
• An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
• In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
• Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

• In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
• However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
• Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
• Such circuits include storage elements that store the values of logic signals.
• Flip-flops can store a one-bit data (information).

Question sources

Computer Science model exam 2 · Question 36

Question 41 | Q00697

Which one of them is a Universal Logic Gates?

A. NAND

B. Ex-OR

C. AND

D. NOT

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer A: NAND

Figure 1.2 on PDF p.7 gives NAND as NOT(A AND B). Tie its inputs to obtain NOT A; NAND followed by that inverter gives AND; applying NAND to inverted inputs gives OR, as can be checked against the same truth tables. Thus NAND alone can build the basic AND/OR/NOT set.

Why each choice is right or wrong

A - Correct: NAND supplies inversion by tying inputs and then AND/OR by composition (p.7, Figure 1.2).

B - Incorrect: XOR alone maps all-zero inputs to zero; it cannot supply the inverter required by Figure 1.2.

C - Incorrect: AND alone also preserves zero and cannot produce NOT using only AND gates.

D - Incorrect: NOT only complements one input; it cannot combine independent input variables into AND/OR.

Primary note proof

Study-note section COA04

> NAND and NOR are each universal: feeding the same input to both inputs constructs NOT; combine complemented gates to construct AND or OR.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Question sources

Computer Science model exam 2 · Question 38

Question 42 | Q00698

Which textbook architecture model stores instructions and data in a shared addressed memory system?

A. Microarchitecture

B. Harvard Architecture

C. Von-Neumann Architecture

D. System Design

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: Von-Neumann Architecture

The Von Neumann model uses a shared addressed memory system for instructions and data. Harvard separates instruction and data memories or paths. Microarchitecture names an internal implementation, while system design is broader than the particular stored-program memory model requested. The revised stem asks about a defining memory relationship rather than claiming that all modern computers use one exclusive organization.

Why each choice is right or wrong

A - Incorrect: Microarchitecture describes internal processor implementation.

B - Incorrect: Harvard separates instruction and data memories or paths.

C - Correct: Von Neumann uses the described shared addressed memory model.

D - Incorrect: System design is broader than this named memory model.

Primary note proof

Study-note section COA02

> In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA02

> Harvard separates instruction and data memories or paths.

Study-note section COA01

> Microarchitecture names the internal processor design.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

> The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be

performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Question sources

Computer Science model exam 2 · Question 39

Question 43 | Q00893

In a design connecting a processor-side bus segment to an unlike peripheral bus segment, which hardware category provides the necessary bridge or interface logic?

A. SCSI

B. PCI

C. Bus bridge/controller logic

D. Multi bus

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Bus bridge/controller logic

Unlike bus segments need bridge or controller logic to adapt their interfaces and coordinate transfers. PCI names a peripheral expansion interconnect, while SCSI names a storage connection; neither alone identifies the adapting component. Multibus names a bus rather than the requested bridge category. The revised topology and component requirement make the interface-logic choice definite without assuming a particular processor bus or expansion standard.

Why each choice is right or wrong

A - Incorrect: SCSI is a storage-connection standard rather than a bridge component.

B - Incorrect: PCI names an expansion interconnect rather than the adapting logic.

C - Correct: Bridge/controller logic connects unlike bus segments.

D - Incorrect: Multibus names a bus rather than the required interface component.

Primary note proof

Study-note section COA32

> PCI is a peripheral expansion interconnect, not a disk protocol; connecting unlike bus segments requires suitable bridge/controller logic.

Study-note section COA32

> SATA is a storage interface; legacy IDE/PATA and SCSI represent other storage connections, while iSCSI carries SCSI commands over IP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 40 | lines 18-23

> 4.2 Bus and Memory Transfers
> A typical computer has many registers, and paths must be provided to transfer information from one register to another. An efficient way for transferring information between registers in a multiple-register configuration is a common bus system. A bus structure consists of a set of common lines, one for each bit of a register. Control signals determine which register the bus selects during each particular register transfer.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

> When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
> • Whether to read or write
> • Device address
> • Starting address of memory block for data
> • Amount of data to be transferred The CPU carries on with other work.
> • Thus DMA controller steals the CPU‟s work of I/O operation.

• The DMA module transfers the entire block of data
• One word at a time, directly to or from memory, without going through CPU.
When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Question sources

Computer Science Exit Exam · 2018 · Question 6

Question 44 | Q00894

Combinational circuit differ from sequential circuit primarily because

A. Combinational circuit cannot be built using logic circuit

B. The output of combinational circuit depends only on the present input, not on past output.

C. .
Sequential circuit have no feedback while combinational circuit do

D. The combinational circuit requires clock signal, but sequential circuit do not

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer B: The output of combinational circuit depends only on the present input, not on past output.

Combinational output depends only on input at the current time (PDF p.10); sequential output also depends on the past and includes memory elements (p.15). This is the primary functional difference, independent of implementation speed.

Why each choice is right or wrong

A - Incorrect: The definition explicitly builds a combinational circuit from gates (p.10), contradicting A.

B - Correct: B restates the present-input-only definition (p.10).

C - Incorrect: Sequential circuits contain memory (p.15); reversing the feedback distinction does not match the combinational definition.

D - Incorrect: The module does not make a clock mandatory for combinational logic; it defines direct current-input dependence (p.10).

Primary note proof

Study-note section COA05

A combinational output depends on present inputs after propagation delay.

Study-note section COA05

A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Study-note section COA05

Clocked sequential circuits update at designated clock events; combinational logic does not inherently require a clock.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 10 | lines 22-27

1.4 Combinational Circuits
• A combinational circuit is an interconnected set of gates whose output at any time is a function only of the input at that time. As with a single gate, the appearance of the input is followed almost immediately by the appearance of the output, with only gate delays. In general terms, a combinational circuit consists of n binary inputs and m binary outputs.

Computer Organization and Architecture module | PDF page 15 | lines 9-16

• In case of combinational circuits, the value of each output depends on the values of signals applied to the inputs.
• However, in case of Sequential Circuits, the values of the outputs depend not only on the present values of the inputs but also on the past behavior of the circuit.
• Sequential circuits consist of a combinational circuit and some memory elements. The memory elements are used to store information about the past.
• Such circuits include storage elements that store the values of logic signals.
• Flip-flops can store a one-bit data (information).

Question sources

Computer Science Exit Exam · 2018 · Question 7

Question 45 | Q00909

Which one of the following is the first operation or step in CPU instruction cycle?

A. Executing the instruction

B. Handling interrupt

C. .
Decoding the instruction

D. Fetching the instruction

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Fetching the instruction

The listed cycle begins with fetching an instruction, then decoding, possible indirect-address access, and execution (PDF p.55). In the fetch micro-operations PC first supplies AR and memory then supplies IR.

Why each choice is right or wrong

A - Incorrect: Execution follows fetch and decode, rather than initiating the cycle (p.55).

B - Incorrect: Interrupt handling is not the first phase in the module's listed ordinary instruction cycle.

C - Incorrect: Decoding is the second listed phase (p.55).

D - Correct: Fetching is explicitly phase 1, with register transfers shown on p.55.

Primary note proof

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 55 | lines 2-8

> 1. Fetch an instruction from memory
> 2. Decode the instruction
> 3. Read the effective address from memory if the instruction has an indirect address
> 4. Execute the instruction
> Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

> I. Fetch the instruction
> Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
> T0: AR ← PC
> Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented

T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1,
INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Question sources

Computer Science Exit Exam · 2018 · Question 22

Question 46 | Q00913

Which one of the following is a computer structural component that provides for communication among CPU, main memory and I/O?

A. System unit

B. Computer network

C. CPU

D. System interconnection

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: System interconnection

The structural-component list defines system interconnection as the mechanism providing communication among CPU, main memory and I/O (PDF p.6). This definition exactly matches the requested role.

Why each choice is right or wrong

A - Incorrect: “System unit” is not the named communication component in the list on p.6.

B - Incorrect: A network is not the internal mechanism named in this structural definition.

C - Incorrect: CPU is one of the three communicating parties, not the interconnection itself (p.6).

D - Correct: The definition of system interconnection matches the entire stem (p.6).

Primary note proof

Study-note section COA01

CPU, main memory, I/O and their interconnection are the major structural parts.

Study-note section COA-S02

A bus is a set of common signal lines used to transfer binary information among connected units.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Question sources

Computer Science Exit Exam · 2018 · Question 26

Question 47 | Q00918

is a data transfer method in computer organization and architecture that allows an I/O device to transfer data directly to or from memory without the involvement of the CPU.

A. Asynchronous data transfer

B. Direct Memory Access (DMA)

C. .
Interrupt-driven transfer

D. Programmed I/O transfer

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Direct Memory Access (DMA)

DMA sends the block directly to/from memory and keeps CPU involvement to setup and completion (PDF pp.107-108). Asynchrony describes timing, whereas interrupt/programmed I/O still make the CPU transfer data.

Why each choice is right or wrong

A - Incorrect: Asynchronous transfer describes independent clocks (p.101), not a bypass path.

B - Correct: DMA words go directly to or from memory without traversing CPU (p.108).

C - Incorrect: The CPU executes transfers under interrupt I/O (p.106).

D - Incorrect: Programmed I/O has CPU status checks and data transfer (p.104).

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer’s motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

> When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
> • Whether to read or write
> • Device address
> • Starting address of memory block for data
> • Amount of data to be transferred The CPU carries on with other work.
> • Thus DMA controller steals the CPU‟s work of I/O operation.
> • The DMA module transfers the entire block of data
> • One word at a time, directly to or from memory, without going through CPU.
> When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Computer Organization and Architecture module | PDF page 101 | lines 2-9

> Asynchronous data transfer
> The internal operation in digital system is synchronized by means of clock pulse supplied by a common pulse generator. Two units such as cpu and an I/O are designed independently of each other, if the register in the interface share a common clock with the cpu registers, the transfer between the two units is called to be synchronous.in most case the internal timing in each unit is independent from the other in that each uses its own private clock for internal

register.in such cases the two units are said to be asynchronous to each other. This approach is widely used in most computers systems.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Question sources

Computer Science Exit Exam · 2018 · Question 31

Question 48 | Q01178

Which instruction class tests a data word or computation status and sets condition information, often without retaining an arithmetic result?

A. Test instructions

B. Branch instructions

C. All of the above

D. Data-transfer instructions

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Test instructions

TEST/CMP-style instructions examine values and update condition information, often discarding an arithmetic result. Branch instructions use control-flow conditions to choose a target; data-transfer instructions copy values. Those are different roles, so the aggregate alternative cannot identify the testing class. The cleaned question retains the recognizable, image-confirmed test-instruction concept while removing displaced OCR debris.

Why each choice is right or wrong

A - Correct: Test instructions examine values and set condition information.

B - Incorrect: Branches choose control-flow targets rather than naming the class that establishes the test condition.

C - Incorrect: Branch and data-transfer instructions do not both have the specified test role, so the aggregate is false.

D - Incorrect: Data-transfer instructions copy values rather than performing the requested condition-setting test.

Primary note proof

Study-note section COA19

> TEST/CMP examine values and set condition information, often without retaining the arithmetic result.

Study-note section COA19

> Branch instructions choose a target, conditionally or unconditionally.

Study-note section COA19

> Data-transfer instructions copy values.

Study-note section COA19

> Arithmetic/logic instructions compute values.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 55 | lines 2-8

1. Fetch an instruction from memory
2. Decode the instruction
3. Read the effective address from memory if the instruction has an indirect address
4. Execute the instruction
Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Computer Organization and Architecture module | PDF page 51 | lines 14-22

Instruction Set Completeness
A computer should have a set of instructions so that the user can construct machine language program to evaluate any function that is known to be computable. The set of instructions are said to be complete if the computer includes a sufficient number of instructions in each of the following categories:
1. Arithmetic, Logical and Shift Instructions.
2. Instructions for moving information to and from memory and processor registers.
3. Program control instructions together with instructions that check status conditions.
4. Input and Output instructions.

Question sources

Computer Science Exit Exam practice questions · Question 39

Question 49 | Q01179

Branch instructions are used to branch to a different set of instructions depending on the decision made.

A. True

B. False

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: True

A branch changes the ordinary sequential instruction path to the branch address. The module explicitly identifies the actual branch address in a branch instruction (PDF p.72), and visually inspected Table 5-2 (p.52) shows both unconditional branch and conditional skip operations. Interpret “depending on the decision” as conditional branching; it is not true that every branch must be conditional.

Why each choice is right or wrong

A - Correct: True under the conditional-branch interpretation: the address field supplies the alternate target (p.72).

B - Incorrect: False would deny that conditional transfers select another instruction sequence; the table’s conditional skip examples illustrate such decisions (p.52).

Primary note proof

Study-note section COA19

> Branch instructions choose a target, conditionally or unconditionally.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 72 | lines 26-32

> Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
> Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Computer Organization and Architecture module | PDF page 55 | lines 2-8

1. Fetch an instruction from memory
2. Decode the instruction
3. Read the effective address from memory if the instruction has an indirect address
4. Execute the instruction
Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 52 | lines 2-6

Table 5-2: Basic computer instructions
Arithmetic, logical and shift instructions provide computational capabilities for processing the type of data that the user may wish to employ. The bulk of the binary information in a digital computer is stored in memory, but all computations are done in processor registers. Therefore, the user must have the capability of moving information between these two units

Question sources

Computer Science Exit Exam practice questions · Question 40

Question 50 | Q01183

Which of the following is not considered as a peripheral device?

A. CPU

B. Keyboard

C. Monitor

D. All of the above

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer A: CPU

Peripherals are external input/output devices such as keyboard and display (PDF p.99). CPU is the central processor structural component (p.6), so it is the only offered non-peripheral.

Why each choice is right or wrong

A - Correct: CPU is the processor, distinct from the external I/O devices (p.6).

B - Incorrect: Keyboard is explicitly a peripheral (p.99).

C - Incorrect: Display/video monitor is explicitly peripheral (p.99).

D - Incorrect: Keyboard and monitor are peripherals, so not all choices satisfy the negative stem.

Primary note proof

Study-note section COA29

> A peripheral performs input, output or storage outside the CPU's execution core; CPU itself is not a peripheral.

Study-note section COA29

> Input units encode external information and output units present processed results.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 99 | lines 3-7

> 7.1 Peripheral (External) Devices
> Input or output devices attached to the computer are called peripherals. Among the most known peripherals some of them are, keyboard display unit and printers.
> Video monitors are the most commonly used peripheral. They consists keyboard as the input device and display unit as an output devices.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

> 7.3 Classification of External devices
> External devices broadly can be classified into three categories:
> Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
> Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
> Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Question sources

Computer Science model exam 2 · Question 4

Question 51 | Q01187

Which one of the following statements is not true about the application of micro-operations?

A. Selective set operation is implemented using OR micro-operation

B. Selective clear operation clears to zero the bit in the first operand only where there are corresponding 1's in the logical operand

C. Clear operation is implemented using Exclusive-OR Micro-operation.

D. None of the above

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: None of the above

Selective set uses A OR B and selective clear zeroes A where B has ones (PDF p.44). For the unqualified clear operation, XOR a register with itself: each bit x XOR x=0, as shown by the equal-input rows of Figure 1.2 on PDF p.7. Thus A-C can all be true and D answers the negative stem.

Why each choice is right or wrong

A - Incorrect: The OR expression and example for selective set are explicit (p.44).

B - Incorrect: The selective-clear definition exactly states zeroing A at B=1 positions (p.44).

C - Incorrect: XOR self-clear is valid because 0 XOR 0 and 1 XOR 1 both equal zero (p.7, visually checked truth table).

D - Correct: None of A-C is false under self-XOR clearing, so D is the exception response.

Primary note proof

Study-note section COA13

> To set selected bits use A OR mask; to clear them use A AND NOT(mask); to toggle use A XOR mask; to retain only selected bits use A AND mask.

Study-note section COA13

> A XOR A clears the entire register.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 44 | lines 16-31

Selective-set: The selective-set operation sets to 1 the bits in register A where there are corresponding 1's in register B. It does not affect bit positions that have 0's in B.
e.g 1010 A Before
1100 B (logic operand)
1110 A after, the corresponding micro operation is A←A V B
Selective-complement: The selective-complement operation complements bits in A where there are corresponding 1's in B. It does not affect bit positions that have 0's in B.
e.g 1010 A before
1100 B (logic operand)
0110 A after , the corresponding logic micro operation A←A B
Selective-clear: The selective-clear operation clears to 0 the bits in A only where there are corresponding 1's in B.
e.g 1010 A before
1100 B (logic operand)
0010 A after
The corresponding logic micro operation is A← A $\wedge^{-}$

Computer Organization and Architecture module | PDF page 7 | lines 3-9

Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Computer Organization and Architecture module | PDF page 38 | lines 8-12

Digital modules are best defined by the registers they contain and the operations that are performed on the data stored in them. The operations executed on data stored in registers are called micro operations. A micro operation is an elementary operation performed on the information stored in one or more registers.
Examples of micro operation are shift, count, clear, and load.

Computer Organization and Architecture module | PDF page 75 | lines 5-13

b. Logical and Bit manipulation Instructions
Logical instructions perform binary operations on strings of bits stored in registers. They are useful for manipulating individual bits or a group of bits that represent binary-coded information. The logical instructions consider each bit of the operands separately and treat it as a Boolean variable. By proper application of the logical instructions it is possible to change bit values, to clear a group of bits or to insert new bit values into the operands stored in registers or memory words. Some logical and bit manipulation instructions are listed in Table 5-8 as follows.
Table 5-8 Some logical and bit manipulation instructions

Question sources

Computer Science model exam 2 · Question 8

Question 52 | Q01190

In textbook discussions of richer CISC instructions, which term names the distance between high-level language constructs and machine operations that those instructions may reduce?

A. Time delay

B. Semantic gap

C. Cost

D. All of the mentioned

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer B: Semantic gap

The semantic gap is the distance between high-level constructs and machine operations. Richer instructions can reduce that gap by expressing elaborate operations directly. Time and cost are separate design concerns, and neither RISC nor CISC guarantees reductions in them across implementations and workloads. The revised question ties the named gap to the supported richer-instruction discussion instead of asserting one universal purpose shared by both approaches.

Why each choice is right or wrong

A - Incorrect: Time delay is a performance measure rather than this conceptual distance.

B - Correct: Semantic gap names the high-level-to-machine-operation distance.

C - Incorrect: Cost is an implementation measure rather than this gap.

D - Incorrect: The question asks for one specific conceptual term.

Primary note proof

Study-note section COA20

> The semantic gap is the distance between high-level constructs and machine operations; reducing it is associated with richer instructions, not a universal shared objective of both design approaches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

> RISC characteristics
> • Relatively few instructions
> • Relatively few addressing modes
> • Memory access limited to load and store instructions
> • All operations done within the register of the CPU

• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 78 | lines 2-9

5.19 CISC Instruction:
CISC refers to Complex Instruction Set Computer
CISC characteristic
• Large number of instructions - typically from 100 to 250 instructions
• Some instructions that perform specialized tasks and are used infrequently
• A large variety of addressing modes - typically from 5 to 20 different modes
• Variable length instruction formats
• Instructions that manipulate operands in memory RISC characteristics

Question sources

Computer Science model exam 2 · Question 11

Question 53 | Q01248

Which register identifies the next instruction to fetch, before its address is copied into the address register for a memory access?

A. Accumulator

B. Instruction register

C. Address register

D. Program counter

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Program counter

The program counter identifies the next instruction to fetch. During fetch, that address can be copied into MAR so memory can perform the access, but MAR's general role is the address of the current memory operation. IR holds the fetched instruction itself and the accumulator holds operands or results. Specifying next instruction and the point before copying separates PC from the later memory-access address.

Why each choice is right or wrong

A - Incorrect: The accumulator holds operands or results.

B - Incorrect: IR holds the fetched instruction itself.

C - Incorrect: MAR holds the current memory-access address after it is loaded.

D - Correct: PC identifies the next instruction to fetch.

Primary note proof

Study-note section COA14

> PC identifies the next instruction to fetch.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

> Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

> 6.2.1 Connection of RAM and CPU
> Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

I. Fetch the instruction
Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Question sources

Computer Science model exam 2 · Question 69

Question 54 | Q01250

Which of the following logic gates has an output value that is identical to its input?

A. NAND

B. NOR

C. Exclusive-OR

D. Buffer

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer D: Buffer

The enabled three-state buffer has output Y=A when control C=1 (PDF p.41). Therefore buffer preserves the input logic value. Figure 1.2 on p.7 gives the other gates' differing two-input truth tables.

Why each choice is right or wrong

A - Incorrect: NAND complements AND; it is not an identity function (p.7).

B - Incorrect: NOR complements OR; it is not an identity function (p.7).

C - Incorrect: XOR depends on both inputs; it equals one input only under an extra fixed-zero assumption (p.7).

D - Correct: An enabled buffer explicitly gives Y=A (p.41).

Primary note proof

Study-note section COA03

> An enabled buffer copies its input.

Study-note section COA03

> AND is 1 only when all inputs are 1; OR is 1 if any input is 1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 41 | lines 12-18

> The bus system can be constructed with three-state gates instead of multiplexers. A three-state gate is a digital circuit that exhibits three states. Two of the states are signals equivalent to logic 1 and 0 as in a conventional gate. The third state is a high- impedance state. The high-impedance state behaves like an open circuit, which means that the output is disconnected and does not have logic significance.
> Output Y=A if C=1
> High Impedance if C=0

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Question sources

Computer Science model exam 2 · Question 71

Question 55 | Q01280

From the labels RISC and CISC alone, which conclusion about processor power efficiency is justified without implementation or workload information?

A. RISC

B. Neither label alone determines power efficiency

C. IANA

D. CISC

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer B: Neither label alone determines power efficiency

RISC and CISC describe instruction-set design tendencies, not universal power rankings. Power efficiency depends on implementation and the work being performed, so neither label alone establishes which processor is more efficient. The option states that supported conclusion. RISC and CISC remain unjustified blanket selections, while IANA does not supply the processor implementation or workload needed to resolve the comparison.

Why each choice is right or wrong

A - Incorrect: RISC does not guarantee greater power efficiency.

B - Correct: The note rejects a guaranteed power ranking from these labels alone.

C - Incorrect: This does not provide a relevant processor implementation for the comparison.

D - Incorrect: CISC does not guarantee greater power efficiency.

Primary note proof

Study-note section COA20

> Neither label guarantees lower power, cost or execution time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

> RISC characteristics
> • Relatively few instructions
> • Relatively few addressing modes
> • Memory access limited to load and store instructions
> • All operations done within the register of the CPU
> • Fixed length, easily decoded instruction format
> • Single cycle instruction execution
> • Hard-wired rather than micro-programmed control
> • A relatively large number of registers in the processor

• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 78 | lines 2-9

5.19 CISC Instruction:
CISC refers to Complex Instruction Set Computer
CISC characteristic
• Large number of instructions - typically from 100 to 250 instructions
• Some instructions that perform specialized tasks and are used infrequently
• A large variety of addressing modes - typically from 5 to 20 different modes
• Variable length instruction formats
• Instructions that manipulate operands in memory RISC characteristics

Computer Organization and Architecture module | PDF page 118 | lines 3-19

A RISC (Reduced Instruction Set Computer) processor pipeline operates in much the same way, although the stages in the pipeline are different. While different processors have different numbers of steps, they have basically variations of these five steps:
1. fetch instructions from memory
2. read registers and decode the instruction
3. execute the instruction or calculate an address
4. access an operand in data memory
5. write the result into a register
The length of the pipeline is dependent on the length of the longest step. Because RISC instructions are simpler than those used in pre-RISC processors (now called CISC, or Complex Instruction Set Computer), they are more conducive to pipelining. While CISC instructions varied in length, RISC instructions are all the same length and can be fetched in a single operation. Ideally, each of the stages in a RISC processor pipeline should take 1 clock cycle so that the processor finishes an instruction each clock cycle and averages one cycle per instruction (CPI). Among the characteristics attributed to RISC is its ability to use an efficient instruction pipeline.

Question sources

Computer Science model exam 2 · Question 101

Question 56 | Q01304

Which of the following is not a type of memory used in an operating system?

A. Cache

B. RAM

C. ROM

D. Ethernet

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: Ethernet is a network connection, not a memory type.

COA23 describes cache as copying memory blocks near the CPU, and COA24 describes RAM technologies plus nonvolatile ROM. COA32 explicitly classifies Ethernet as connecting network devices rather than memory. Thus cache, RAM and ROM are memory-related choices, while Ethernet belongs to communications. The phrase used in an operating system does not convert a network technology into storage hardware.

Why each choice is right or wrong

A - Incorrect: Cache stores copies of memory blocks and is a genuine level of the memory hierarchy.

B - Incorrect: RAM is main-memory storage, not a communication technology excluded from the memory category.

C - Incorrect: ROM is nonvolatile memory even though its normal access and modification rules differ from RAM.

D - Correct: Ethernet connects network devices and is explicitly distinguished from memory in the note.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA24

> ROM and flash retain contents without power; erasable/programmed variants mean “ROM can never be modified” is too broad.

Study-note section COA32

> Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

> 6.2 Main Memory
> The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 81 | lines 6-16

> Static RAM:
> • a bit of data is stored using the state of a flip-flop.
> • Applying power is enough (no need for refreshing).

• Retains value indefinitely, as long as it is kept powered.
• Mostly uses to create cache memory of CPU
• Faster and more expensive than DRAM.
ROM
ROM is used for storing programs that are permanently resident in the computer and for tables of constants that do not change in value once the production of the computer is completed The ROM portion of main memory is needed for storing an initial program called bootstrap loader, which is to start the computer software operating when power is

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 80 | lines 5-11

DYNAMIC RAM (DRAM)
is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Question sources

Operating Systems practice questions · Question 22

Question 57 | Q01318

Which of the following is not a type of storage device used in an operating system?

A. Hard disk drive (HDD)

B. Solid-state drive (SSD)

C. Tape drive

D. Ethernet cable

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: Ethernet cable is a communication medium.

COA28 describes HDD magnetic storage, SSD nonvolatile semiconductor storage and tape for sequential archives. It then explicitly distinguishes an Ethernet cable as a communication medium rather than a disk or RAM. The question asks for the item outside the storage-device category, so the cable is the supported exclusion regardless of which filesystem or OS uses the other devices.

Why each choice is right or wrong

A - Incorrect: An HDD stores persistent magnetic patterns in addressed blocks, so it is a storage device.

B - Incorrect: An SSD stores data in nonvolatile semiconductor storage, satisfying the storage-device category.

C - Incorrect: A tape drive provides sequential storage suitable for streaming and archives, so it is not excluded.

D - Correct: An Ethernet cable carries communication signals rather than functioning as a persistent storage device.

Primary note proof

Study-note section COA28

> An HDD stores magnetic patterns on tracks divided into sectors; reads/writes access addressed blocks while platters rotate.

Study-note section COA28

> SSDs use nonvolatile semiconductor storage without mechanical seek.

Study-note section COA28

Optical media use lasers; tape is suited to sequential streaming and archives.

Study-note section COA28

An Ethernet cable is a communication medium, not a disk or RAM.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Computer Organization and Architecture module | PDF page 98 | lines 7-16

6.4.2 Magnetic Tape
• Magnetic Tape is a plastic ribbon which is usually ½ inch or ¼ inch wide & 50 to 2400 feet long.
• It is coated with iron-oxide material.
• It is similar to the tape of audio cassettes of tape recorders. Data is stored as binary digits.
• Data is accessed sequentially so searching becomes difficult.
• Tape systems use the same reading and recording techniques as disk systems..
• Magnetic tape was the first kind of secondary memory.It is still widely used as the lowest-cost, slowest-speed member of the memory hierarchy.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.

• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

Operating Systems practice questions · Question 36

Question 58 | Q01465

Which generation of computers used vacuum tubes as their primary electronic component?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer A: First generation: vacuum tubes.

Under the familiar generation convention in COA38, vacuum tubes identify the first generation. Transistors belong to the second, integrated circuits to the third, and microprocessor/VLSI-era personal computers to the fourth. This question asks for the defining electronic component in that convention, so the mapping provides a direct answer without needing exact boundary dates for every historical machine.

Why each choice is right or wrong

A - Correct: The first generation is the conventional vacuum-tube era specified in the complete note.

B - Incorrect: The second generation is associated with transistors rather than vacuum tubes as its defining component.

C - Incorrect: The third generation is associated with integrated circuits rather than the vacuum-tube era.

D - Incorrect: The fourth generation is associated with microprocessors/VLSI rather than vacuum tubes as its primary component.

Primary note proof

Study-note section COA38

The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Question sources

Operating Systems practice questions · Question 13

Question 59 | Q01466

Which generation of computers utilized transistors for electronic circuitry?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer B: Second generation: transistors.

COA38 maps the second generation to transistor-based circuitry. The surrounding categories distinguish the first vacuum-tube era, third integrated-circuit era and fourth microprocessor/VLSI era. Within this conventional classification, the transistor clue therefore selects the second generation. The answer concerns the defining technology, rather than claiming transistors ceased to exist when integrated circuits appeared.

Why each choice is right or wrong

A - Incorrect: The first generation is conventionally identified with vacuum tubes, not transistor-based circuitry.

B - Correct: The note explicitly associates transistor circuitry with the second generation of computers.

C - Incorrect: Integrated circuits define the third-generation category, rather than the discrete-transistor era requested here.

D - Incorrect: Microprocessor/VLSI-era computers define the fourth category, rather than the transistor-generation convention.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

> DYNAMIC RAM (DRAM)
> is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
> STATIC RAM (SRAM)
> In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 30 | lines 3-19

> 3.1 Integrated Circuits
> • Integrated circuit (IC) is the basic building block of digital circuits.
> • An integrated circuit is a small silicon semiconductor crystal, called a chip.
> • The various gates are interconnected inside the chip to form the required circuit.
> • As the technology of ICs has improved, the number of gates that can be put in a single chip has increased.
> ✓ Small-scale integration (SSI) devices contain several (usually less than 10) independent gates in a single package.
> ✓ Medium-scale integration (MSI) devices contain approximately 10 to 200 gates in a single package.
> E.g To form decoders, adders, and registers.
> ✓ Large-scale integration (LSI) devices contain between 200 and a few thousands gates in a single package.
> Eg. processors, memory chips, and programmable modules.
> ✓ Very-large-scale integration (VLSI) devices contain thousands of gates in a single package.
> E.g large memory arrays and complex microcomputer chips.

Question sources

Operating Systems practice questions · Question 14

Question 60 | Q01467

Under the course's conventional generation summary, around the transition into which generation did high-level programming languages become prominent?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer B: Second Generation

The course associates high-level languages becoming prominent with the transition into the second generation. That is a conventional historical summary, rather than a claim that every language was first introduced on one sharply defined boundary. The revised verb preserves the generation-history question while avoiding an unsupported universal introduction era. The other options refer to different stages in the familiar circuitry-based chronology.

Why each choice is right or wrong

A - Incorrect: The note instead places prominence around the transition into the second era.

B - Correct: This matches the explicitly stated historical convention.

C - Incorrect: The third era is associated with integrated circuits rather than this transition.

D - Incorrect: The fourth era is associated with microprocessor/VLSI personal computers.

Primary note proof

Study-note section COA38

> High-level languages became prominent around the transition into the second; introduction dates do not fit every generation boundary.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

> RISC characteristics
> • Relatively few instructions
> • Relatively few addressing modes
> • Memory access limited to load and store instructions

• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Question sources

Operating Systems practice questions · Question 15

Question 61 | Q01468

Which generation of computers introduced the concept of stored program architecture?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer A: First generation.

COA38 directly places stored-program machines in the first era. The stored-program clue is different from the transistor, integrated-circuit and microprocessor clues defining later generations. Under the historical convention supplied by this note, the concept therefore belongs to the first generation, even though later computers also retained stored-program operation.

Why each choice is right or wrong

A - Correct: The note explicitly places the rise of stored-program machines in the first generation.

B - Incorrect: The second generation retained stored programs but is not where the note places their introduction.

C - Incorrect: The third generation is the integrated-circuit era, later than the stated stored-program introduction.

D - Incorrect: The fourth generation is the microprocessor era, much later than the stated first-era development.

Primary note proof

Study-note section COA38

Stored-program machines arose in the first era.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Question sources

Operating Systems practice questions · Question 16

Question 62 | Q01469

Which generation of computers saw the development of integrated circuits?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer C: Third generation: integrated circuits.

COA38 associates the third generation with integrated circuits. The first is identified by vacuum tubes, the second by transistors and the fourth by microprocessor/VLSI-era personal computers. Reading development in the question as the conventional adoption/defining-technology category gives third generation; it does not require assuming that every invention date perfectly matches textbook boundaries.

Why each choice is right or wrong

A - Incorrect: The first-generation convention identifies vacuum tubes rather than integrated circuits as the defining technology.

B - Incorrect: The second-generation convention identifies transistor circuitry, preceding the integrated-circuit generation category.

C - Correct: Integrated circuits are the defining third-generation technology in the supplied historical convention.

D - Incorrect: The fourth generation is the later microprocessor/VLSI category rather than the initial integrated-circuit category.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Computer Organization and Architecture module | PDF page 30 | lines 3-19

> 3.1 Integrated Circuits
> • Integrated circuit (IC) is the basic building block of digital circuits.
> • An integrated circuit is a small silicon semiconductor crystal, called a chip.
> • The various gates are interconnected inside the chip to form the required circuit.
> • As the technology of ICs has improved, the number of gates that can be put in a single chip has increased.
> ✓ Small-scale integration (SSI) devices contain several (usually less than 10) independent gates in a single package.
> ✓ Medium-scale integration (MSI) devices contain approximately 10 to 200 gates in a single package.
> E.g To form decoders, adders, and registers.
> ✓ Large-scale integration (LSI) devices contain between 200 and a few thousands gates in a single package.
> Eg. processors, memory chips, and programmable modules.
> ✓ Very-large-scale integration (VLSI) devices contain thousands of gates in a single package.
> E.g large memory arrays and complex microcomputer chips.

Question sources

Operating Systems practice questions · Question 17

Question 63 | Q01470

A historical computer uses magnetic-core memory and transistor-based circuitry rather than vacuum tubes or integrated circuits. Under the conventional taxonomy, which generation describes its circuitry?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer B: Second Generation

Magnetic-core memory alone does not identify a generation because it spanned more than one era. The added circuitry condition supplies the distinguishing feature: the conventional taxonomy associates transistor-based machines with the second generation. Vacuum tubes identify the first era, integrated circuits the third and microprocessor/VLSI personal computers the fourth. This version retains the magnetic-core context and states which technology determines the classification.

Why each choice is right or wrong

A - Incorrect: First-generation circuitry uses the vacuum-tube convention.

B - Correct: Transistor circuitry identifies the second generation.

C - Incorrect: Third-generation circuitry uses the integrated-circuit convention.

D - Incorrect: The fourth era is associated with microprocessor/VLSI systems.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Study-note section COA38

> Magnetic-core memory spanned more than one era, so “witnessed use” cannot identify exactly one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

6.2 Main Memory
The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

DYNAMIC RAM (DRAM)
is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Operating Systems practice questions · Question 18

Question 64 | Q01472

Which generation conventionally introduced integrated-circuit circuitry, contributing to improved speed and performance over earlier discrete-transistor systems?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer C: Third Generation

Several generations improved speed, so improvement alone is not a unique historical marker. The revised question identifies integrated circuits as the distinguishing technology. Under the stated convention, integrated circuits identify the third generation, following transistor-based second-generation systems. Performance improvement remains a contextual trend rather than a guarantee that every individual later machine outperformed every earlier model.

Why each choice is right or wrong

A - Incorrect: The first era is associated with vacuum tubes.

B - Incorrect: The second era is associated with discrete transistors.

C - Correct: The third era is associated with integrated circuits.

D - Incorrect: The fourth era is associated with microprocessor/VLSI systems.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Study-note section COA38

> Integration generally reduced individual component dimensions while capacity, density and processor performance grew; these are trends rather than guarantees for every model.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

> RISC characteristics
> • Relatively few instructions
> • Relatively few addressing modes
> • Memory access limited to load and store instructions
> • All operations done within the register of the CPU
> • Fixed length, easily decoded instruction format
> • Single cycle instruction execution
> • Hard-wired rather than micro-programmed control
> • A relatively large number of registers in the processor
> • Use overlapped register windows to speed up procedure call and return
> • Efficient instruction pipeline
> • Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of

memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Question sources

Operating Systems practice questions · Question 20

Question 65 | Q01473

Which generation of computers introduced the use of microprocessors?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer D: Fourth generation: microprocessors.

The conventional mapping in COA38 identifies the fourth generation with microprocessor/VLSI-era computers. Vacuum tubes, transistors and integrated circuits identify the preceding first, second and third categories respectively. Thus the microprocessor clue directly selects the fourth generation under the note's convention; the question does not ask for an exact launch date or individual processor model.

Why each choice is right or wrong

A - Incorrect: Vacuum tubes define the first-generation convention, rather than the later microprocessor era.

B - Incorrect: The transistor-based second-generation category precedes the note's microprocessor/VLSI category.

C - Incorrect: Third generation denotes integrated-circuit computers; microprocessors identify the subsequent conventional category.

D - Correct: The complete note explicitly associates microprocessor/VLSI-era computers with the fourth generation.

Primary note proof

Study-note section COA38

The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Question sources

Operating Systems practice questions · Question 21

Question 66 | Q01474

Which generation of computers is characterized by the development of personal computers?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer D: Fourth generation: personal computers.

COA38 groups personal computers with the fourth-generation microprocessor/VLSI era. The first three alternatives instead correspond to vacuum tubes, transistors and integrated circuits. The wording characterized by asks for that conventional defining association, so fourth generation is supported without making a universal claim that no personal or individual-use machine could exist in any earlier period.

Why each choice is right or wrong

A - Incorrect: The first era is characterized by vacuum tubes in this convention, not microprocessor-based personal computers.

B - Incorrect: The second era is the transistor category rather than the personal-computer era identified here.

C - Incorrect: The third era is associated with integrated circuits rather than the note's personal-computer category.

D - Correct: The note places microprocessor/VLSI-era personal computers in the fourth-generation historical convention.

Primary note proof

Study-note section COA38

> The familiar generation convention associates first with vacuum tubes, second with transistors, third with integrated circuits and fourth with microprocessor/VLSI-era personal computers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Question sources

Operating Systems practice questions · Question 22

Question 67 | Q01491

A stack organized computer uses instruction of

A. Zero addressing

B. Two addressing

C. Index addressing

D. Indirect addressing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Zero addressing

For stack arithmetic the operands are implicit on top of stack, so ADD and MUL require no address field (PDF p.69). This is the zero-address organization. PUSH/POP still use explicit addresses; the answer concerns arithmetic operation instructions.

Why each choice is right or wrong

A - Correct: Zero-address ADD/MUL is explicit (p.69).

B - Incorrect: Two-address instructions instead explicitly name register/memory operands; stack arithmetic does not (pp.67-69).

C - Incorrect: Index addressing is an effective-address mechanism, not the stack operand-count format.

D - Incorrect: Indirect addressing locates an address through memory (p.72); it is not the stack arithmetic format.

Primary note proof

Study-note section COA16

> Zero-address stack ADD uses two implicit top-of-stack operands.

Study-note section COA17

> A stack is last-in, first-out.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 69 | lines 15-23

> 5.15.4 Zero-Address Instruction
> A stack-organized computer does not use an address field for instructions ADD and MUL.
> The PUSH and POP instructions, however, need an address field to specify the operand that communicates with the stack. The following program shows how X = (A+B) * (C+D) will be written for a stack-organized computer (TOS stands for top-of-stack)
> PUSH A TOS← A, PUSH B TOS ←B, ADD TOS ←(A + B),

PUSH C TOS ←C
PUSH D TOS ←D, ADD TOS ←(C + D) , MUL TOS ←(C + D) (A + B),
POP X M[X] ←TOS.

Computer Organization and Architecture module | PDF page 72 | lines 2-19

Immediate Mode: In this mode the operand is specified in the instruction itself. In other words, an immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction. Immediate mode instructions are useful for initializing registers to a constant value.
When the address field specifies a processor register, the instruction is said to be in the register mode.
Register Mode: In this mode the operands are in registers that reside within the CPU. The particular register is selected from a register field in the instruction. A k-bit field can specify any of the 2k registers.
Register Indirect Mode: In this mode the instruction specifies a register in the CPU whose contents give the address of the operand in memory. In other words, the selected register contains the address of the operand rather than the operand itself. Before using a register indirect mode instruction, the programmer must ensure that the memory address of the operand is placed in the processor register with a previous instruction. A reference to the register is then equivalent to specifying a memory address. The advantage of the register indirect mode instruction is that the address field of the instruction uses fewer bits to select a register than would have been required to specify a memory address directly.

Question sources

Mock exam 1 · Question 7

Question 68 | Q01492

An n-bit microprocessor has -

A. No answer is correct

B. n-bit program counter

C. n-bit address register

D. n-bit instruction register

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: None of the named register widths necessarily follows.

COA38 explicitly states that an n-bit processor label does not force its PC, IR or address register to be n bits. Those widths concern different architectural quantities. Since the stem supplies only the processor-width label and no specific design, none of the three claimed register widths is guaranteed. The no-answer-is-correct alternative is therefore the supported response.

Why each choice is right or wrong

A - Correct: None of the three specific register-width claims is implied by the processor-width label alone.

B - Incorrect: Program-counter width is not forced to equal the processor's n-bit label in every architecture.

C - Incorrect: Address-register width can differ from processor word width; n bits is not guaranteed.

D - Incorrect: Instruction-register width depends on instruction encoding rather than necessarily matching the processor-width label.

Primary note proof

Study-note section COA38

> An n-bit processor label does not force its PC, IR or address register to be n bits.

Study-note section COA12

> Word width, data-bus width, instruction width and address width need not match.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

> 3.9 Registers
> • A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
> • A register is a digital circuit used within the CPU to store one or more bits of data.
> • An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
> • In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
> • Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

> 6.2.1 Connection of RAM and CPU
> Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

> The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

> An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Question sources

Mock exam 1 · Question 8

Question 69 | Q01493

In which of the following term the performance of cache memory is measured?

A. Copy ratio

B. Data ratio

C. Hit ratio

D. Cache ratio

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Hit ratio

Cache performance is explicitly measured by hit ratio, hits/(hits+misses), in PDF p.85. A hit means the requested word is found in cache (p.84), making this a direct measure of cache availability of requested data.

Why each choice is right or wrong

A - Incorrect: Copy ratio is not the performance quantity defined on p.85.

B - Incorrect: Data ratio is not the named hit/miss measure.

C - Correct: Hit ratio and its formula are explicitly provided (p.85).

D - Incorrect: “Cache ratio” is not the quantity’s stated name.

Primary note proof

Study-note section COA23

> Hit ratio=hits/accesses; miss ratio=1−hit ratio.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 85 | lines 2-8

> • The performance of cache memory is frequently measured in terms of a quantity called hit ratio
> Hit ratio = hit / (hit+miss)
> • The basic characteristic of cache memory is its fast access time
> • Therefore, very little or no time must be wasted when searching the words in the cache
> • The transformation of data from main memory to cache memory is referred to as a mapping process.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

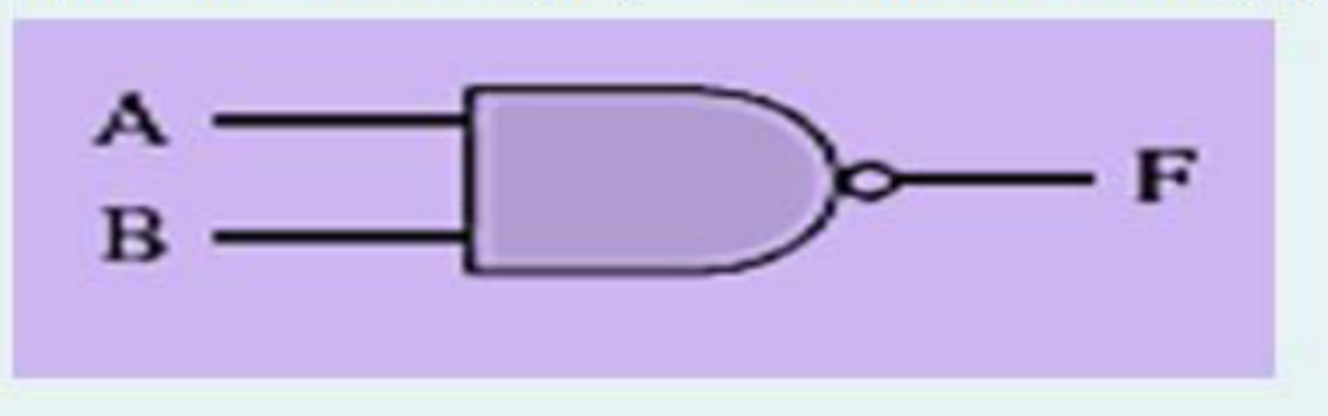
> 6.3 Cache Memory
> Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
> Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
> The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
> • The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
> • When CPU needs to access memory, the cache is examined
> • If the word is found in the cache, it is read from the fast memory
> • If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
> • When the CPU refers to memory and finds the word in cache, it is said to produce a hit
> • Otherwise, it is a miss

Question sources

Mock exam 1 · Question 9

Question 70 | EX2018Q100

Question number 100
What is the name of logic gate based on the following graphical symbol?



A. XOR

B. NOR

C. NAND

D. AND

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: NAND

The diagram shows an AND-shaped gate with a small output circle. This is a NAND gate; the circle indicates inversion, so F=NOT(A AND B).

Why each choice is right or wrong

A - Incorrect: XOR uses the OR-like body with an extra input curve, unlike the source symbol (p.7).

B - Incorrect: NOR uses an OR-shaped body with an output circle, not the flat-input AND body (p.7).

C - Correct: The AND body plus inversion circle is NAND (p.7).

D - Incorrect: AND lacks the output inversion circle visible in the question.

Primary note proof

Study-note section COA03

> A bubble at a gate output indicates inversion: an AND-shaped symbol with a bubble is NAND.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal. The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Question sources

Computer Science Exit Exam · 2018 · Question 100

Question 71 | SM0013

An interface that has multiple lines connecting the input/output module and peripheral device and assures multiple bits to be transferred at the same time is:

A. Bus interface

B. Parallel interface

C. One line interface

D. Serial interface

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Parallel interface

Parallel registers read/write bits simultaneously (PDF p.35); shift registers transfer serially (p.36), and a common bus supplies a line for each bit (p.40). Multiple lines carrying bits at the same time match a parallel interface by this distinction.

Why each choice is right or wrong

A - Incorrect: A bus describes a collection of lines; it does not specify serial versus parallel transfer as precisely as B (p.40).

B - Correct: Simultaneous multiple-bit transfer corresponds to the parallel operation defined on p.35.

C - Incorrect: One line cannot carry the described multiple separate bit signals simultaneously in this binary-line model.

D - Incorrect: Serial shifting transfers the stream sequentially rather than all bits simultaneously (p.36).

Primary note proof

Study-note section COA31

Serial sends bits over time; parallel transfers multiple bits on multiple lines, without guaranteeing higher real-world speed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 35 | lines 3-11

3.9 Registers
• A register is a group of flip-flops with each flip- flop capable of storing one bit of information.
• A register is a digital circuit used within the CPU to store one or more bits of data.
• An n-bit register has a group of n flip-flops and is capable of storing n bits of binary information.
• In addition to Flip-Flops, it may have combinational gates that perform certain data processing.
• Two basic types of registers are commonly used: parallel registers and shift registers.

Computer Organization and Architecture module | PDF page 36 | lines 2-9

Shift Register
• A shift register accepts and/or transfers information serially.
• Data are input only to the leftmost flip-flop.
• Shift registers can be used to interface to serial I/O devices.
• With each clock pulse, data are shifted to the right, one position, and the rightmost bit is transferred out.
• In addition, they can be used within the ALU to perform logical shift and rotate functions.

Computer Organization and Architecture module | PDF page 40 | lines 18-23

4.2 Bus and Memory Transfers
A typical computer has many registers, and paths must be provided to transfer information from one register to another. An efficient way for transferring information between registers in a multiple-register configuration is a common bus system. A bus structure consists of a set of common lines, one for each bit of a register. Control signals determine which register the bus selects during each particular register transfer.

Question sources

MOE Computer Science Exit Exam · Question 4-1

National Computer Science Exit Exam · Question 96

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 228

Question 72 | SM0033

Which one of the following is a two-level cache having an internal and external caches?

A. Multilevel cache

B. Single level cache

C. Split cache

D. Unified cache

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer A: Multilevel cache.

A two-level hierarchy has more than one cache level, so multilevel is the matching category. COA26 distinguishes L1/L2 hierarchy levels from split/unified instruction-data organization and from internal/external placement. Describing one cache as internal and another as external does not make the pair split or unified; those terms answer a different classification question.

Why each choice is right or wrong

A - Correct: Two cache levels constitute a multilevel hierarchy, regardless of their internal or external placement.

B - Incorrect: Single-level cache contradicts the explicitly stated presence of two cache levels in the hierarchy.

C - Incorrect: Split describes separate instruction and data caches, not an internal-plus-external level arrangement.

D - Incorrect: Unified describes combined instruction/data caching, not how many internal or external levels exist.

Primary note proof

Study-note section COA26

> L1/L2 describe hierarchy levels; split cache means separate instruction/data caches and unified means combined.

Study-note section COA26

> Internal/external location is another classification, not synonymous with split/unified.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

> 6.3 Cache Memory
> Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.

Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 79 | lines 3-13

6.1 Memory Hierarchy:
A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Question sources

MOE Computer Science Exit Exam · Question 10-2

National Computer Science Exit Exam · Question 26

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 229

Question 73 | SM0051

Which statement is NOT correct about computer architecture and organization?

A. An architecture can remain for a long time but its organization can change

B. Manufacturers offer computers with the same architecture but different organization

C. An architecture can be used for a long time encompassing different computer models

D. Different organizations cannot implement the same architecture

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Different organizations cannot implement the same architecture

Architecture describes programmer-visible behavior, while organization describes internal implementation. Several internal implementations can preserve the same architecture, allowing it to span models and remain stable while implementation details change. The false statement denies that possibility. The other choices describe compatible consequences of the distinction, without requiring a maximum architecture lifetime or a universal implementation-change rate.

Why each choice is right or wrong

A - Incorrect: Implementation can change while architecture stays stable.

B - Incorrect: Different organizations can implement one architecture.

C - Incorrect: One architecture can encompass different computer models over time.

D - Correct: This denies the supported possibility of different implementations of one architecture.

Primary note proof

Study-note section COA01

> Different organizations can implement the same architecture; neither definition imposes a maximum lifetime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 5 | lines 12-20

> Computer architecture: refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory.
> Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

Question sources

MOE Computer Science Exit Exam · Question 15-2

National Computer Science Exit Exam · Question 59

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 230

Question 74 | SM0059

_____ is a set of attributes that can be seen by a program and can have direct impact on the logical execution of a program.

A. Arrangement

B. Organization

C. Flow control

D. Architecture

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Architecture

Programmer-visible attributes that directly affect logical execution are the module's exact definition of computer architecture (PDF p.5). Organization instead comprises operational units and interconnections realizing those specifications.

Why each choice is right or wrong

A - Incorrect: Arrangement is not the defined programmer-visible attribute set.

B - Incorrect: Organization describes implementation units and interconnections (p.5).

C - Incorrect: Flow control is a function, not the definition of this attribute set.

D - Correct: Architecture exactly matches the definition and examples (p.5).

Primary note proof

Study-note section COA01

> Architecture describes programmer-visible behavior: instructions, registers, data types, address spaces and I/O conventions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 5 | lines 12-20

> Computer architecture: refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory.
> Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

Question sources

MOE Computer Science Exit Exam · Question 17-2

National Computer Science Exit Exam · Question 20

National Computer Science Exit Exam

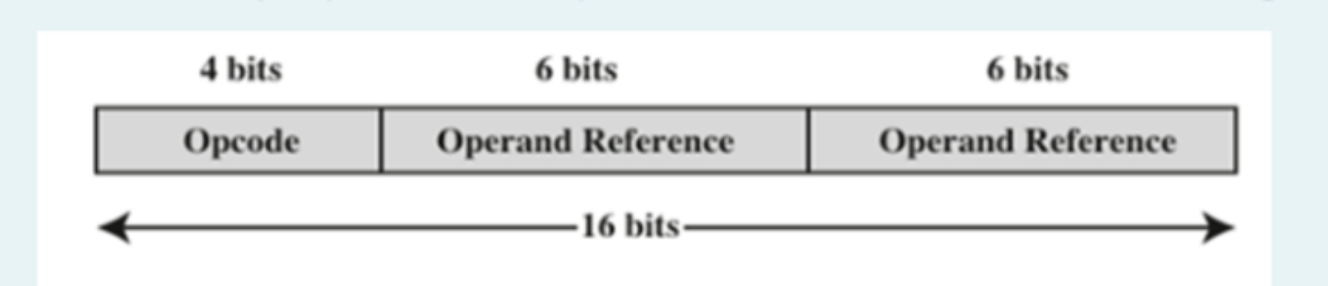
MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 231

Question 75 | SM0063

What is the purpose of an opcode that is as indicated in the figure below?
4 bits 6 bits 6 bits

[opcode] Operant etree [| Opsrand Rates|
<_ 16 bits -ccccxxcxor



A. It holds the destination address of an operand

B. It determines the operation to be done

C. It holds the source address of an operand

D. It holds the next instruction reference

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: It determines the operation to be done

The inspected diagram divides 16 bits into a 4-bit opcode and two 6-bit operand references. The module defines the operation part as specifying what operation is performed (PDF p.47); reference fields identify operands rather than changing that opcode role. Four opcode bits can distinguish up to 16 operation codes.

Why each choice is right or wrong

A - Incorrect: Destination reference belongs to an operand/address field, not the operation field (p.47).

B - Correct: Operation selection is the explicit opcode definition (p.47).

C - Incorrect: Source reference belongs to an operand/address field.

D - Incorrect: The module tracks the next instruction through PC (p.70), not this operation field.

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Study-note section COA16

> A 4-bit opcode permits at most 16 encodings, not necessarily 16 implemented instructions.

Study-note section COA16

> A 6-bit operand-reference field has 64 encodings; what they designate depends on the machine.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

> An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

MOE Computer Science Exit Exam · Question 18-2

Question 76 | SM0091

Which one of the following registers holds the address of the next instruction to be fetched from memory?

A. Memory address register

B. Memory buffer register

C. Program counter

D. Instruction buffer register

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Program counter

PC holds the address of the instruction to execute next (PDF p.70). During fetch it is copied to AR and the instruction bits then enter IR while PC increments (p.55). Thus PC is the persistent next-instruction tracker.

Why each choice is right or wrong

A - Incorrect: MAR can temporarily receive the fetch address, but PC is the register that tracks the next instruction across cycles (p.55).

B - Incorrect: The module describes MDR as memory data storage (p.82), not the next-instruction address tracker.

C - Correct: Directly matches the PC definition and fetch transfers (pp.55,70).

D - Incorrect: An instruction buffer would hold instruction content; the module identifies IR for that role (p.48), not as the next-address tracker.

Primary note proof

Study-note section COA14

> PC identifies the next instruction to fetch.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be

executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

I. Fetch the instruction
Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1,
INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 82 | lines 13-17

6.2.1 Connection of RAM and CPU
Data transfer between the main memory and the CPU register takes place through two registers namely MAR (memory address register) and MDR (memory data register). If MAR is k bits long and MDR is n bits long, the main memory unit can contain up to 2k addressable locations.

Question sources

MOE Computer Science Exit Exam · Question 26-1

National Computer Science Exit Exam · Question 70

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 233

Question 77 | SM0201

Which one of the following is NOT an I/O command?

A. Status command

B. Output data command

C. Control command

D. Issue command

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: Issue command

The module enumerates four I/O command kinds: control, test, read and write (PDF pp.104-105). “Status command” is interpreted as the test/status-query kind, and “output data” as write. “Issue” is an action performed by CPU, not one of the four command kinds.

Why each choice is right or wrong

A - Incorrect: Status-query corresponds to the test command, which checks status conditions (p.104).

B - Incorrect: Output data corresponds to the write command (p.105).

C - Incorrect: Control command activates a peripheral and specifies its action (p.104).

D - Correct: Issue is not a command category in the four-type list (pp.104-105).

Primary note proof

Study-note section COA29

> Control commands start actions, test/status commands inspect readiness or errors, read commands bring device data toward the system, and write/output commands send system data to the device.

Study-note section COA29

> “Issue” describes sending a command, not a standard fourth command category.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 104 | lines 17-27

7.6 I/O commands
To execute an I/O related instruction, the CPU issues an address, specifying the particular I/O module and external device and an I/O command. Four types of I/O commands can be received by the I/O module when it is addressed by the CPU. They are
• A control command: is used to activate a peripheral and tell what to do. Example: a magnetic tape may be directed to rewind or move forward a record.
• A test command: is used to test various status conditions associated with an I/O module and its peripherals. The CPU wants to know the interested peripheral for use. It also wants to know the most recent I/O operation is completed and if any errors have occurred.

Computer Organization and Architecture module | PDF page 105 | lines 2-6

• A read command: it causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer. The CPU then gets the data items by requesting I/O module to place it on the data bus.
• A write command: it causes the I/O module to take an item of data from the data bus and subsequently transmit the data item to the peripheral.

Computer Organization and Architecture module | PDF page 103 | lines 2-9

Status reporting:. I/O module can report with the status signals.common used status signals are BUSY or READY. Various other status signals may be used to report various error conditions.
Address recognition:
just as each memory word has an address, there is address associated with every I/O device. Thus I/O module must be recognized with a unique address for each peripheral it controls. The I/O module must also be able to perform device communication. This communication involves commands,status information, and data. Some of the essentials tasks are listed

Computer Organization and Architecture module | PDF page 101 | lines 10-16

7.4 Input/output Problems
Wide variety of peripherals and delivering different amounts of data per second Work at different speeds Send/receive data in different formats all slower than CPU and RAM. Hence I/O modules are used as a solution.
Input/output Module: It is the entity within a computer that is responsible for the control of one or more external devices and for the exchange of data between those devices and main memory and/or CPU.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 1-1

Computer Science Exit Exam · Sene 2016 · Question 31-3

Computer Science Exit Exam practice collection · Question 243

Question 78 | SM0230

The user who programs a computer in machine or assembly language must be aware of the following EXCEPT:

A. The register set

B. The type of data supported by the instructions

C. The analysis model

D. The memory structure

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: The analysis model

The machine-language programmer's architecture includes instruction formats, addressing modes, instruction sets and register organization (PDF p.57); visible data types and memory addressing are also named (p.5). An "analysis model" is not part of that architectural list.

Why each choice is right or wrong

A - Incorrect: Register organization is explicitly part of the programmer view (p.57).

B - Incorrect: Supported data representation is explicitly architectural (p.5).

C - Correct: Analysis model is absent from the programmer-facing attribute list; it is the unmatched option.

D - Incorrect: Memory addressing is part of the architecture (p.5), under the intended visible-memory interpretation.

Primary note proof

Study-note section COA02

> Machine-language programming depends on the register set, supported types and memory/addressing structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU

performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 5 | lines 12-20

Computer architecture: refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory.
Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 10-1

Computer Science Exit Exam practice collection · Question 244

Question 79 | SM0239

Which part of the computer system is responsible for converting machine-processed data to formats understandable by the user?

A. Communication components

B. Output unit

C. Storage unit

D. Input unit

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Output unit

Computational results must be recorded/displayed for users (PDF p.99); human-readable screen/display/printer devices provide that communication (p.100). This is the output role.

Why each choice is right or wrong

A - Incorrect: Communication devices exchange data with remote devices; the category is not specifically user-readable presentation (p.100).

B - Correct: Output devices record/display computed results in meaningful form for users (p.99).

C - Incorrect: Storage retains information; it is not the presentation role (p.91).

D - Incorrect: Input enters programs/data into the system, the reverse direction (p.99).

Primary note proof

Study-note section COA29

> Input units encode external information and output units present processed results.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 99 | lines 13-23

> The input/output subsystem of a computer, referred to as I/O, provides an efficient mode of communication between the central system and the outside environment. Programs and data must be entered into computer memory for processing and results obtained from computations must be recorded or displayed for users. A computer serves no useful purpose without the ability to receive information from an outside source and to transmit results in a meaningful form. I/O operations are accomplished through a wide assortment of external devices that provide a means of exchanging data between the external environment and the computer. An external device attaches to the computer by a link to an I/O module. The link is used to exchange control, status, and data between the I/O module and the external device. An external device connected to an I/O module also called Interface is often referred to as a peripheral device or simply a peripheral.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

> 7.3 Classification of External devices
> External devices broadly can be classified into three categories:
> Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
> Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
> Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 91 | lines 8-15

> 6.4 Secondary memory
> Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
> • It stores information that is not necessarily in current use.
> • It is slower and having higher capacity than primary memory.
> • This kind of memory is large, slow and inexpensive.
> • It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
> • Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 13-1

Computer Science Exit Exam practice collection · Question 240

Question 80 | SM0242

In the usual memory hierarchy, which listed tier has the shortest access latency and greatest cost per stored bit?

A. Register

B. Magnetic disc

C. Main memory

D. Cache memory

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer A: Register

The usual speed and cost-per-bit hierarchy places registers above cache, main RAM and magnetic disk. Registers therefore have the shortest access latency and highest cost per bit among the choices. Faster tiers are generally smaller, so this statement does not compare the total purchase price of a complete device. Cache is fast, but remains below the register tier in this teaching hierarchy.

Why each choice is right or wrong

A - Correct: Registers are the fastest and most costly-per-bit tier among these choices.

B - Incorrect: Magnetic disk is a slower, high-capacity tier with lower cost per bit.

C - Incorrect: Main memory lies below registers and cache in the usual speed/cost hierarchy.

D - Incorrect: Cache is fast but below registers in the usual hierarchy.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA23

> Faster levels are generally smaller and more expensive per bit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of

memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 58 | lines 4-11

In the programming examples of the previous chapter, we have shown that memory locations are needed for storing pointers, counters, return addresses, temporary results and partial product during multiplication. Having to refer to memory locations for such applications is time consuming because memory access is the most time consuming operation in a computer. It is more convenient and more efficient to store these intermediate values in processor register. When a large number of registers are included in the CPU, It is most efficient to connect them through a common bus system. The registers communicate with each other not only for direct data transfers, but also while performing various micro operations.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 14-1

Computer Science Exit Exam practice collection · Question 241

Question 81 | SM0277

For a legacy internal PC disk connection using parallel ATA, which listed interface name matches that connection?

A. SCSI

B. iSCSI

C. IDE

D. USB

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: IDE

The note identifies legacy IDE/PATA as a storage connection, so IDE matches the specified parallel ATA context. SCSI is a different storage interface, iSCSI carries SCSI over IP, and USB commonly connects peripherals including external storage. Naming the legacy internal parallel interface resolves the missing era and connection-role constraints without asserting that one interface is universally standard for all PCs and all storage devices.

Why each choice is right or wrong

A - Incorrect: SCSI is a different storage-connection standard.

B - Incorrect: iSCSI transports SCSI commands over IP rather than using the specified local parallel interface.

C - Correct: IDE corresponds to the stated legacy PATA connection.

D - Incorrect: USB is a different peripheral interconnect, not parallel ATA.

Primary note proof

Study-note section COA32

> SATA is a storage interface; legacy IDE/PATA and SCSI represent other storage connections, while iSCSI carries SCSI commands over IP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 25-2

Computer Science Exit Exam practice collection · Question 245

Question 82 | SM0295

In the simple CPU-master memory-bus model, which bus carries the values read from and written to memory?

A. Control bus

B. Data bus

C. Uni-bus

D. Address bus

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Data bus

The data bus carries memory-transfer values. Reads move values from memory toward the CPU, and writes move values in the opposite direction. Address signals select a location and control signals coordinate timing and read/write operations. The revised stem asks about payload rather than bidirectionality in general, since aggregate control can also contain signals in both directions. Uni-bus names a bus arrangement rather than this payload-signal category.

Why each choice is right or wrong

A - Incorrect: Control signals coordinate the transfers rather than carrying their values.

B - Correct: The data bus carries read and write values.

C - Incorrect: This is not the address/data/control payload-signal category requested.

D - Incorrect: Address signals choose the memory location.

Primary note proof

Study-note section COA12

Address signals select a location; data signals carry values; control signals coordinate read/write, timing, requests and acknowledgements.

Study-note section COA12

In the simple CPU-master model, addresses flow from CPU to memory, while data travels in either direction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 105 | lines 2-6

• A read command: it causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer. The CPU then gets the data items by requesting I/O module to place it on the data bus.
• A write command: it causes the I/O module to take an item of data from the data bus and subsequently transmit the data item to the peripheral.

Computer Organization and Architecture module | PDF page 102 | lines 20-24

I/O module must have the capability to engage in communication with the CPU and external device. Thus CPU communication involves
Command decoding: The I/O module accepts commands from the CPU carried on the control bus.
Data: data are exchanged between the CPU and the I/O module over data bus

Computer Organization and Architecture module | PDF page 103 | lines 2-9

Status reporting:. I/O module can report with the status signals.common used status signals are BUSY or READY. Various other status signals may be used to report various error conditions.
Address recognition:
just as each memory word has an address, there is address associated with every I/O device. Thus I/O module must be recognized with a unique address for each peripheral it controls. The I/O module must also be able to perform device communication. This communication involves commands,status information, and data. Some of the essentials tasks are listed

Question sources

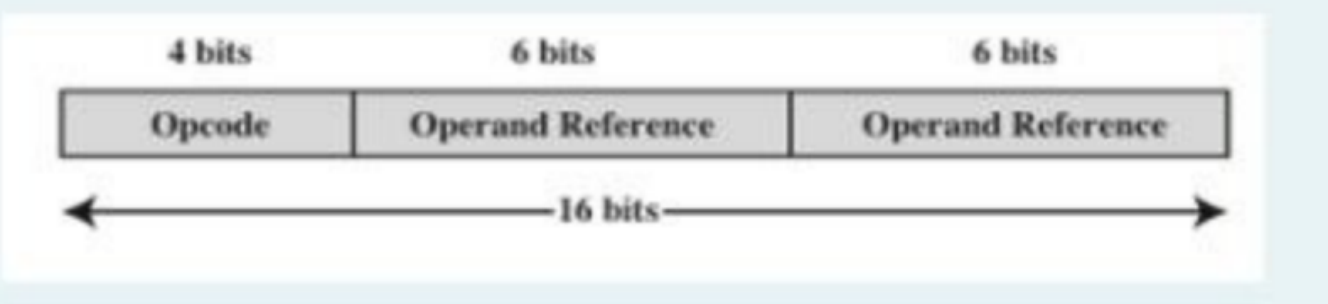
Computer Science Exit Exam · Sene 2016 · Question 31-2

Computer Science Exit Exam practice collection · Question 242

Question 83 | SS077

What is the purpose of an opcode that is as indicated in the figure below?
4 bits 6 bits 6 bits

<j o bits --_____»



A. It determines the operation to be done

B. It holds the destination address of an operand

C. It holds the next instruction reference

D. It holds the source address of an operand

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: It determines the operation to be done

The instruction has a 4-bit Opcode followed by two 6-bit Operand Reference fields. Opcode selects the operation.

Why each choice is right or wrong

A - Correct: Matches the operation-code definition (p.47).

B - Incorrect: A destination reference is an operand field, not the opcode.

C - Incorrect: Next-instruction location is tracked by PC (p.70), not this operation field.

D - Incorrect: Source address belongs to an operand reference, not the operation code (p.47).

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

> An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The

most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

National Computer Science Exit Exam · Question 80

National Computer Science Exit Exam

Question 84 | AM0704

Which of the following is a common way of communicating with I/O devices?

A. Memory-mapped I/O

B. Direct memory access (DMA)

C. Interrupt-driven I/O

D. All of the above

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: All of the above

Memory-mapped I/O shares memory and device address space (PDF p.105); DMA transfers blocks directly (p.107); interrupt I/O requests CPU service when ready (p.106). They are all supported I/O mechanisms, though memory mapping is an addressing choice and the others are transfer/control techniques.

Why each choice is right or wrong

A - Incorrect: Memory-mapped I/O is explicitly described (p.105).

B - Incorrect: DMA is explicitly described (p.107).

C - Incorrect: Interrupt-driven I/O is explicitly described (p.106).

D - Correct: All three are legitimate mechanisms under the broad “communicating” wording.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA31

> Memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable load/store operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 105 | lines 7-18

> 7.8 I/O Mapping
> When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
> • Memory mapped I/O
> • Devices and memory share an address space
> • I/O looks just like memory read/write
> • No special commands for I/O
> • Large selection of memory access commands available
> • Isolated I/O
> • Separate address spaces
> • Need I/O or memory select lines
> • Special commands for I/O and Limited set

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer’s motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Question sources

Computer Science MCQ practice collection · Question 704

Question 85 | AM0705

In memory-mapped I/O, I/O devices are mapped into the:

A. Main memory address space

B. Cache memory address space

C. Virtual memory address space

D. Secondary memory address space

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer A: Main memory address space

Memory-mapped I/O makes devices and memory share an address space, with I/O treated like memory reads/writes (PDF p.105). The relevant address space is the memory address space, not a separate cache or secondary-storage space.

Why each choice is right or wrong

A - Correct: Devices share the memory address space (p.105).

B - Incorrect: The module does not map device registers into a separate cache address space.

C - Incorrect: The mapping definition does not depend on a process virtual address space.

D - Incorrect: Secondary-storage addresses are not the shared CPU memory addresses described on p.105.

Primary note proof

Study-note section COA31

Memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable load/store operations.

Study-note section COA31

Isolated/port-mapped I/O has a separate port space and special instructions such as IN/OUT.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 105 | lines 7-18

7.8 I/O Mapping
When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
• Memory mapped I/O
• Devices and memory share an address space
• I/O looks just like memory read/write
• No special commands for I/O
• Large selection of memory access commands available
• Isolated I/O
• Separate address spaces
• Need I/O or memory select lines
• Special commands for I/O and Limited set

Question sources

Computer Science MCQ practice collection · Question 705

Question 86 | AM0706

Which technique allows data to be transferred between an I/O device and memory without involving the CPU?

A. Memory-mapped I/O

B. Direct memory access (DMA)

C. Interrupt-driven I/O

D. I/O polling

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Direct memory access (DMA)

DMA transfers each word directly to/from memory without passing through CPU (PDF p.108). “Without involving” must mean during the actual word transfers, since CPU setup and completion involvement are explicit on the same page.

Why each choice is right or wrong

A - Incorrect: Memory mapping specifies how I/O is addressed, not a CPU-bypass data engine (p.105).

B - Correct: DMA directly transfers the block with CPU only at beginning/end (p.108).

C - Incorrect: The CPU still executes the interrupt-driven data transfer (p.106).

D - Incorrect: Polling keeps CPU checking device status and doing transfers (p.104).

Primary note proof

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA30

> DMA does not eliminate CPU involvement in configuration, ownership, address mapping or completion.

Study-note section COA31

> Memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable load/store operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 107 | lines 10-17

> 7.10 Direct Memory Access (DMA)
> Interrupt driven and programmed I/O require active CPU intervention.
> Transfer rate is limited
> CPU is tied up DMA is the solution for these problems
> Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer’s motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 108 | lines 2-12

> When the CPU wishes to read or write a block of data, it issues a command to the DMA module and gives following information: CPU tells DMA controller:
> • Whether to read or write
> • Device address
> • Starting address of memory block for data
> • Amount of data to be transferred The CPU carries on with other work.
> • Thus DMA controller steals the CPU‟s work of I/O operation.
> • The DMA module transfers the entire block of data
> • One word at a time, directly to or from memory, without going through CPU.
> When the transfer is complete, DMA controller sends interrupt when finished Thus CPU is involved only at the beginning and at the end of the transfer.

Computer Organization and Architecture module | PDF page 105 | lines 7-18

7.8 I/O Mapping
When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
• Memory mapped I/O
• Devices and memory share an address space
• I/O looks just like memory read/write
• No special commands for I/O
• Large selection of memory access commands available
• Isolated I/O
• Separate address spaces
• Need I/O or memory select lines
• Special commands for I/O and Limited set

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Question sources

Computer Science MCQ practice collection · Question 706

Question 87 | AM0707

Which method is used to handle I/O operations that require immediate attention from the CPU?

A. Memory-mapped I/O

B. Direct memory access (DMA)

C. Interrupt-driven I/O

D. I/O polling

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Interrupt-driven I/O

In interrupt-driven I/O the device requests CPU service when ready; CPU can work on other instructions until then (PDF p.106). The priority-interrupt discussion on that page services the higher-priority simultaneous request first. This matches event-driven attention, without guaranteeing zero response latency.

Why each choice is right or wrong

A - Incorrect: Memory mapping selects addresses and does not itself notify the CPU (p.105).

B - Incorrect: DMA offloads data movement (p.107); it is not the general CPU-service notification technique.

C - Correct: The device interrupts CPU requesting service when ready (p.106).

D - Incorrect: Polling requires the CPU to discover readiness through repeated checks (p.104).

Primary note proof

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA31

> Interrupt priority selects among simultaneous requests, using schemes such as polling, daisy chains or priority hardware.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 105 | lines 7-18

7.8 I/O Mapping
When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
• Memory mapped I/O
• Devices and memory share an address space
• I/O looks just like memory read/write
• No special commands for I/O
• Large selection of memory access commands available
• Isolated I/O
• Separate address spaces
• Need I/O or memory select lines
• Special commands for I/O and Limited set

Computer Organization and Architecture module | PDF page 107 | lines 10-17

7.10 Direct Memory Access (DMA)
Interrupt driven and programmed I/O require active CPU intervention.
Transfer rate is limited
CPU is tied up DMA is the solution for these problems
Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of

actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Question sources

Computer Science MCQ practice collection · Question 707

Question 88 | AM0708

Which I/O technique involves the CPU continuously checking the status of an I/O device for data transfer completion?

A. Memory-mapped I/O

B. Direct memory access (DMA)

C. Interrupt-driven I/O

D. I/O polling

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: I/O polling

Programmed I/O explicitly makes CPU periodically check I/O status until completion, because the module does not interrupt CPU (PDF p.104). This is the polling behavior described by D.

Why each choice is right or wrong

A - Incorrect: Memory mapping is the address arrangement and does not require repeated status checks (p.105).

B - Incorrect: DMA permits CPU other work while moving a block (p.108).

C - Incorrect: Interrupt I/O notifies CPU when ready instead of requiring periodic checking (p.106).

D - Correct: CPU checks status bits periodically until completion (p.104).

Primary note proof

Study-note section COA30

Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 104 | lines 2-16

Programmed I/O :With Programmed I/O, data are exchanged between the CPU and the I/O module. The CPU executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command and transferring data. When CPU issues a command to I/O module, it must wait until I/O operation is complete. If the CPU is faster than I/O module, there is wastage of CPU time. The I/O module does not take any further action to alert CPU. That is it doesn't interrupt CPU. Hence it is the responsibility of the CPU to periodically check the status of the I/O module until it finds that the operation is complete. The sequences of actions that take place with programmed I/O are:
• CPU requests I/O operation
• I/O module performs operation
• I/O module sets status bits
• CPU checks status bits periodically
• I/O module does not inform CPU directly
• I/O module does not interrupt CPU

Computer Organization and Architecture module | PDF page 106 | lines 8-20

with the higher priority first.
Interrupt Driven I/O.
Using Program-controlled I/O requires continuous involvement of the processor in the I/O activities. It is desirable to avoid wasting processor execution time. An alternative is for the CPU to issue an I/O command to a module and then go on other work. The I/O module will then interrupt the CPU requesting service when it is ready to exchange data with the CPU. The CPU will then execute the data transfer and then resumes its former processing. Based on the use of interrupts, this technique improves the utilization of the processor. With Interrupt driven I/O, the CPU issues a command to I/O module and it does not wait until I/O operation is complete but instead continues to execute other instructions. When I/O module has completed its work it interrupts the CPU. An interrupt is more than a simple mechanism for coordinating I/O transfers. In a general sense, interrupts enable transfer of control from one program to another to be initiated by an event that is external to a computer. Execution of the

Computer Organization and Architecture module | PDF page 107 | lines 10-17

7.10 Direct Memory Access (DMA)
Interrupt driven and programmed I/O require active CPU intervention.
Transfer rate is limited
CPU is tied up DMA is the solution for these problems
Direct Memory Access is capabilities provided by some computer bus architectures that allow data to be sent directly from an attached device (such as a disk drive) to the memory on the computer's motherboard. The microprocessor (CPU) is freed from involvement with the data transfer, thus speeding up overall computer operation.

Computer Organization and Architecture module | PDF page 105 | lines 7-18

7.8 I/O Mapping
When the CPU, main memory, and I/O module share a common bus two modes of addressing are possible.
• Memory mapped I/O
• Devices and memory share an address space
• I/O looks just like memory read/write
• No special commands for I/O
• Large selection of memory access commands available
• Isolated I/O
• Separate address spaces
• Need I/O or memory select lines
• Special commands for I/O and Limited set

Question sources

Computer Science MCQ practice collection · Question 708

Question 89 | AM0709

Which standard I/O interface is commonly used in UNIX-like operating systems?

A. ASCII

B. USB

C. RS-232

D. POSIX

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: POSIX.

COA32 identifies POSIX as a software interface standard with open/read/write/close operations. That is the software I/O interface sense appropriate to UNIX-like operating systems. ASCII is an encoding, while USB and RS-232 describe hardware interconnect/interface roles. Their availability on a UNIX-like machine does not make them the standard OS software interface requested by the alternatives.

Why each choice is right or wrong

A - Incorrect: ASCII defines character encoding, not the UNIX-like software interface for opening, reading and writing resources.

B - Incorrect: USB is a peripheral interconnect, not the software API standard identified by this OS-level wording.

C - Incorrect: RS-232 defines a serial electrical/interface standard rather than the UNIX-like software I/O API.

D - Correct: POSIX specifies software interface operations including open, read, write and close for this context.

Primary note proof

Study-note section COA32

POSIX is a software interface standard with operations such as open/read/write/close; ASCII is a character encoding.

Study-note section COA32

USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 112 | lines 3-19

A data communication processor is an I/O processor that distribute and collect data from many remote terminals connected through telephone and other communication lines it is a specialized I/O processor designed to communicate directly with data communication network. The most striking difference between I/O processor and data communication processor is in the way processor communicate with the I/O devices.an I/O processor communicate with the peripherals through common I/O bus that is comprised of many data and control lines. All peripherals share common bus and use it to transfer information to and from I/O processor. A data communication processor communicates with each terminal through a single pairs of wires.
Data can be transmitted between two points in three different modes: simplex, half duplex

and full duplex.
A simplex line carries information in one direction only. This mode is seldom used in data communication because the receiver cannot communicate to the transmitter with indicate the occurrence of error examples of simplex transmission are radio and television broadcasting. Half duplex transmission system is one that have capable of transmitting in both direction but, data can transmitted in only one direction at a time. A pair of wire is needed for mode. A full duplex transmission can send and receive data in both directions simultaneously. This

Question sources

Computer Science MCQ practice collection · Question 709

Question 90 | AM0710

Which listed interface is a traditional point-to-point serial electrical/interface standard?

A. USB

B. Ethernet

C. RS-232

D. HDMI

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: RS-232

RS-232 specifies a traditional point-to-point serial electrical/interface standard. USB also uses serial communication, so serial alone cannot distinguish them. The revised requirement identifies the traditional point-to-point interface category rather than treating every serial link as equivalent. Ethernet connects network devices and HDMI carries digital audio and video. Their connection roles differ from the interface class requested.

Why each choice is right or wrong

A - Incorrect: USB is a serial peripheral interconnect, but not the traditional point-to-point electrical standard requested.

B - Incorrect: Ethernet provides networking connectivity.

C - Correct: RS-232 matches the traditional point-to-point serial interface category.

D - Incorrect: HDMI carries digital audio and video.

Primary note proof

Study-note section COA32

USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Study-note section COA32

HDMI carries digital video and audio.

Study-note section COA32

Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 101 | lines 2-9

Asynchronous data transfer
The internal operation in digital system is synchronized by means of clock pulse supplied by a common pulse generator. Two units such as cpu and an I/O are designed independently of each other, if the register in the interface share a common clock with the cpu registers, the transfer between the two units is called to be synchronous.in most case the internal timing in each unit is independent from the other in that each uses its own private clock for internal register.in such cases the two units are said to be asynchronous to each other. This approach is widely used in most computers systems.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

7.3 Classification of External devices
External devices broadly can be classified into three categories:
Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Computer Organization and Architecture module | PDF page 36 | lines 2-9

Shift Register
• A shift register accepts and/or transfers information serially.
• Data are input only to the leftmost flip-flop.
• Shift registers can be used to interface to serial I/O devices.
• With each clock pulse, data are shifted to the right, one position, and the rightmost bit is transferred out.
• In addition, they can be used within the ALU to perform logical shift and rotate functions.

Computer Organization and Architecture module | PDF page 112 | lines 3-19

A data communication processor is an I/O processor that distribute and collect data from many remote terminals connected through telephone and other communication lines it is a specialized I/O processor designed to communicate directly with data communication network. The most striking difference between I/O processor and data communication processor is in the way processor communicate with the I/O devices.an I/O processor

communicate with the peripherals through common I/O bus that is comprised of many data and control lines. All peripherals share common bus and use it to transfer information to and from I/O processor. A data communication processor communicates with each terminal through a single pairs of wires.
Data can be transmitted between two points in three different modes: simplex, half duplex and full duplex.
A simplex line carries information in one direction only. This mode is seldom used in data communication because the receiver cannot communicate to the transmitter with indicate the occurrence of error examples of simplex transmission are radio and television broadcasting.
Half duplex transmission system is one that have capable of transmitting in both direction but, data can transmitted in only one direction at a time. A pair of wire is needed for mode.
A full duplex transmission can send and receive data in both directions simultaneously. This

Question sources

Computer Science MCQ practice collection · Question 710

Question 91 | AM0711

Which standard I/O interface is widely used for connecting external storage devices, such as hard drives and flash drives?

A. SATA

B. USB

C. Ethernet

D. HDMI

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: USB.

COA32 explicitly describes USB as a peripheral interconnect used by external storage. This directly matches the stem's external hard-drive and flash-drive examples. SATA is described more generally as a storage interface, Ethernet connects network devices, and HDMI carries video/audio. The external peripheral examples therefore make USB the supported conventional choice among these options.

Why each choice is right or wrong

A - Incorrect: SATA is a storage interface, but USB is the note's explicit match for the external-peripheral examples.

B - Correct: The note explicitly lists external storage among USB peripheral connections, matching the supplied examples.

C - Incorrect: Ethernet connects network devices rather than being the named external hard-drive/flash-drive peripheral interconnect.

D - Incorrect: HDMI carries digital video and audio, rather than providing the external-storage interface requested.

Primary note proof

Study-note section COA32

> USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Study-note section COA32

> HDMI carries digital video and audio.

Study-note section COA32

> Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

> 6.4.1 Magnetic Disk
> • Magnetic disks are the foundation of external memory on virtually all computer systems.
> • A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
> • More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
> • It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
> • We can randomly access the data.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

> 7.3 Classification of External devices
> External devices broadly can be classified into three categories:
> Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
> Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
> Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Question sources

Computer Science MCQ practice collection · Question 711

Question 92 | AM0712

Which listed interface is specifically a storage-drive interface, rather than a general peripheral interconnect, networking interface or audiovisual link?

A. SATA

B. USB

C. Ethernet

D. HDMI

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer A: SATA

SATA is specifically a storage interface. USB supplies general peripheral connectivity, Ethernet connects network devices and HDMI carries digital audio/video. The connection role distinguishes these alternatives; a vague high-speed description would not, because several interfaces can be fast in suitable versions. The question therefore asks for a storage-specific interface category without assuming that speed alone uniquely identifies it.

Why each choice is right or wrong

A - Correct: SATA is the storage-specific interface in the list.

B - Incorrect: USB provides a different peripheral-interconnect role.

C - Incorrect: Ethernet serves networking rather than this drive connection.

D - Incorrect: HDMI carries audiovisual data rather than this storage link.

Primary note proof

Study-note section COA32

> SATA is a storage interface; legacy IDE/PATA and SCSI represent other storage connections, while iSCSI carries SCSI commands over IP.

Study-note section COA32

> USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Study-note section COA32

> HDMI carries digital video and audio.

Study-note section COA32

> Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 40 | lines 18-23

> 4.2 Bus and Memory Transfers
> A typical computer has many registers, and paths must be provided to transfer information from one register to another. An efficient way for transferring information between registers in a multiple-register configuration is a common bus system. A bus structure consists of a set of common lines, one for each bit of a register. Control signals determine which register the bus selects during each particular register transfer.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

> 6.4.1 Magnetic Disk
> • Magnetic disks are the foundation of external memory on virtually all computer systems.
> • A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
> • More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
> • It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
> • We can randomly access the data.

Question sources

Computer Science MCQ practice collection · Question 712

Question 93 | AM0713

Which standard I/O interface is primarily used for audio and video transmission between devices?

A. SATA

B. USB

C. Ethernet

D. HDMI

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer D: HDMI.

COA32 explicitly identifies HDMI as carrying digital video and audio. That directly matches the primary transmission purpose asked here. SATA serves storage, USB serves general peripherals, and Ethernet serves network connections. Other links can transport encoded media in broader systems, but the stem asks the primary interface role, for which HDMI is the specific supported choice.

Why each choice is right or wrong

A - Incorrect: SATA is identified as a storage interface rather than the primary audiovisual connection in this list.

B - Incorrect: USB is a general peripheral interconnect; the note specifically assigns the audiovisual role to HDMI.

C - Incorrect: Ethernet is a network connection rather than the specifically identified digital audiovisual interface.

D - Correct: HDMI is explicitly described as carrying digital video and audio between devices.

Primary note proof

Study-note section COA32

> HDMI carries digital video and audio.

Study-note section COA32

> USB is a serial peripheral interconnect used by keyboards and external storage; RS-232 specifies a traditional point-to-point serial electrical/interface standard.

Study-note section COA32

> Ethernet connects network devices; it is not memory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 100 | lines 9-17

> 7.3 Classification of External devices
> External devices broadly can be classified into three categories:
> Human readable: suitable for communicating with the computer user. Examples: Screen, keyboard, video display terminals (VDT) and printers.
> Machine readable: suitable for communicating with equipment. Examples: magnetic disk & tapes systems, Monitoring and control, sensors and actuators which are used in robotics.
> Communication: These devices allow a computer to exchange data with remote devices, which may be machine readable or human readable. Examples: Modem, Network Interface Card (NIC)

Question sources

Computer Science MCQ practice collection · Question 713

Question 94 | AM0714

Which technique divides instruction processing into stages so that different stages work on successive instructions during the same cycle?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer A: Pipelining

Pipelining overlaps stages of successive instructions: one may be fetched while another is decoded or executed. Superscalar execution instead concerns issuing multiple instructions to functional units in one cycle; it can coexist with a pipeline but is not the definition of the staged organization. Caching supplies nearby copies of memory blocks, and branch prediction guesses future control flow.

Why each choice is right or wrong

A - Correct: Pipelining divides processing into stages and overlaps those stages across successive instructions.

B - Incorrect: Caching retains memory-block copies; it is not the staged instruction organization described.

C - Incorrect: Superscalar architecture provides multiple-issue execution resources rather than defining stage overlap itself.

D - Incorrect: Branch prediction guesses direction or targets, rather than dividing instruction processing into pipeline stages.

Primary note proof

Study-note section COA35

> A pipeline overlaps stages of different instructions; it primarily improves throughput.

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Question sources

Computer Science MCQ practice collection · Question 714

Question 95 | AM0715

Which technique improves the performance of a processor by storing frequently accessed data closer to the processor?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Caching

Caching keeps copies of memory blocks near the CPU, exploiting temporal and spatial locality to reduce many memory-access delays. Pipelining overlaps instruction stages, superscalar execution provides multiple-issue resources, and branch prediction guesses control flow. These mechanisms may coexist, but the stated nearby data-copy storage role specifically identifies caching.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps instruction stages rather than creating a data buffer (p.116).

B - Correct: Cache is explicitly the fast intermediate RAM buffer that supplies hits (p.84).

C - Incorrect: Superscalar execution issues multiple instructions to functional units; it is not a memory-copy storage tier.

D - Incorrect: Branch prediction guesses control-flow direction or targets; it does not name a nearby memory-block buffer.

Primary note proof

Study-note section COA23

> Cache copies memory blocks near the CPU.

Study-note section COA23

> Temporal locality reuses recent data; spatial locality uses nearby locations.

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Study-note section COA36

Branch prediction guesses direction/target and continues speculatively; wrong guesses discard work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Question sources

Computer Science MCQ practice collection · Question 715

Question 96 | AM0716

Which architecture allows multiple instructions to be executed in parallel by having multiple execution units?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer C: Superscalar architecture.

COA36 defines superscalar execution as issuing multiple instructions to multiple functional units in a cycle when dependencies permit. This matches the multiple execution units in the stem. COA35 distinguishes that multiple-issue capability from ordinary pipelining, which overlaps stages. Caching supplies reused memory data and branch prediction speculates about control flow; neither names the multiple-issue architecture.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages but does not by itself provide multiple-instruction issue to several execution units.

B - Incorrect: Caching retains nearby copies of memory blocks rather than defining parallel instruction issue.

C - Correct: Superscalar execution uses multiple functional units to issue several independent instructions in a cycle.

D - Incorrect: Branch prediction guesses future control flow, rather than naming the architecture with multiple execution units.

Primary note proof

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Study-note section COA35

> A pipeline overlaps stages of different instructions; it primarily improves throughput.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 716

Question 97 | AM0717

Which technique predicts the outcome of conditional branches to improve instruction execution in a processor?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Branch prediction.

COA36 says branch prediction guesses a branch's direction/target and continues speculatively, discarding work when the guess is wrong. Predicting the outcome of a conditional branch is exactly that role. Pipelining overlaps processing stages, caching retains reusable data, and superscalar execution issues several instructions; those mechanisms can coexist with prediction but do not replace its definition.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps instruction stages; it does not itself predict the outcome of a conditional branch.

B - Incorrect: Caching retains reusable memory contents, rather than selecting the predicted direction of conditional control flow.

C - Incorrect: Superscalar execution provides multiple issue; it does not name the conditional-outcome prediction technique.

D - Correct: Branch prediction guesses branch direction or target so speculative execution can continue before resolution.

Primary note proof

Study-note section COA36

> Branch prediction guesses direction/target and continues speculatively; wrong guesses discard work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 116 | lines 2-6

> 8.3 Pipelining
> Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The

decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 717

Question 98 | AM0718

Which technique improves memory access performance by organizing memory into hierarchical levels?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Virtual memory

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Caching

Cache is the fast memory-block-copy tier used to reduce the cost of repeated or nearby CPU accesses. Virtual memory instead supplies translated address spaces and can use backing storage for nonresident pages. Pipelining and superscalar execution organize instruction processing, so neither names the memory-hierarchy buffer requested. The answer follows the storage role rather than a general claim that only one mechanism can improve performance.

Why each choice is right or wrong

A - Incorrect: Pipelining divides computational tasks into stages (p.116), not memory storage levels.

B - Correct: Cache supplies the fast tier in the hierarchy and avoids main-memory access on hits (pp.79,84).

C - Incorrect: Superscalar execution concerns multiple instruction issue, not a memory-hierarchy storage tier.

D - Incorrect: Virtual memory supplies an apparent larger memory space (p.90), not the fast-cache-hit mechanism.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA23

> Cache copies memory blocks near the CPU.

Study-note section COA27

> Virtual memory gives processes translated address spaces and protection; backing storage can hold nonresident pages.

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
> Figure 6.1 memory hierarchy
> • Design constraints: How much? How fast? How expensive?
> • Faster access time, greater cost per bit
> • Greater capacity, smaller cost per bit,
> • Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

> 6.3 Cache Memory
> Cache memory is a small, high-speed RAM buffer located between the CPU and main

memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 90 | lines 9-23

Concept of Virtual Memory
A programmer can write a program which requires more memory space than the capacity of the main memory. Such a program is executed by virtual memory technique.
The program is stored in the secondary memory. The memory management unit (MMU) transfers the currently needed part of the program from the secondary memory to the main memory for execution.
This to and fro movement of instructions and data (parts of a program) between the main memory and the secondary memory is called Swapping.
Address Space And Memory Space.
Virtual address is the address used by the programmer and the set of such addresses is called the address space or virtual memory.
An address in main memory is called a location or physical address. The set of such locations in main memory is called the memory space or physical memory.
CPU generated logical address consisting of a logical page number plus the location within that page (x).

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Question sources

Computer Science MCQ practice collection · Question 718

Question 99 | AM0719

Which processor technique issues multiple independent instructions to multiple functional units in one clock cycle, when dependencies permit?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Instruction-level parallelism

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer C: Superscalar architecture

Superscalar execution issues multiple instructions in a cycle when the machine has suitable resources and the instructions' dependencies permit it. Pipelining overlaps stages of different instructions but need not issue several at once. Caching addresses memory access, while instruction-level parallelism is the broader opportunity exploited by several mechanisms. The revised stem asks about multiple issue rather than incorrectly claiming that wider instruction supply necessarily reduces individual memory latency.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages but does not by itself require multiple issue.

B - Incorrect: Caching retains memory-block copies and is not the multiple-issue technique.

C - Correct: Superscalar execution issues multiple instructions to multiple units in a cycle.

D - Incorrect: Instruction-level parallelism is the broader property exploited rather than this specific issue mechanism.

Primary note proof

Study-note section COA36

> Superscalar execution issues multiple instructions to multiple functional units in a cycle when dependencies permit.

Study-note section COA35

> "Fetch/decode one instruction while executing another" describes pipelining, whereas issuing several instructions in one cycle requires multiple-issue resources.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 116 | lines 2-6

> 8.3 Pipelining
> Pipelining is an implementation technique where multiple instructions are overlapped in

execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 114 | lines 2-10

The following are important points about parallel processing:
• A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Question sources

Computer Science MCQ practice collection · Question 719

Question 100 | AM0720

Which technique translates source or intermediate code into target code during program execution, before the translated portion runs?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Just-in-time (JIT) compilation

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Just-in-time (JIT) compilation

JIT compilation translates code while the program is running, often compiling an intermediate representation before executing that portion. This timing distinguishes it from ahead-of-time translation. Pipelining overlaps processing stages, caching retains copies of data or instructions, and superscalar execution issues multiple instructions to execution resources. Those mechanisms do not name runtime source-to-target compilation.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages of executing instructions; it does not translate high-level or intermediate code.

B - Incorrect: Caching retains data or instruction copies; it does not perform the compilation described in the stem.

C - Incorrect: Superscalar architecture issues multiple instructions; it does not translate a source language into target code.

D - Correct: JIT is the only offered compiler; it translates during program execution before the compiled portion executes.

Primary note proof

Study-note section COA02

> JIT compilation happens during program execution, often translating intermediate code.

Study-note section COA02

> Translation before execution alone does not distinguish JIT from an ahead-of-time compiler.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 720

Question 101 | AM0721

Which technique improves performance by predicting the next set of instructions to be executed based on past execution patterns?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Branch prediction

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Branch prediction.

Predicting which instruction stream follows a branch is the direction/target prediction described in COA36. Using past patterns is consistent with that prediction role, whereas pipeline overlap, cached memory copies and multiple instruction issue address different mechanisms. Prediction remains speculative: a wrong guess discards work rather than changing the program's required architectural result.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages and requires instruction supply, but it is not itself next-path prediction.

B - Incorrect: Caching retains copies for reuse; a cached instruction is not a prediction of future control flow.

C - Incorrect: Superscalar issue increases concurrent instruction execution rather than identifying the upcoming branch path.

D - Correct: Predicting branch direction and targets identifies the likely future instruction stream for speculative execution.

Primary note proof

Study-note section COA36

> Branch prediction guesses direction/target and continues speculatively; wrong guesses discard work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the

instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 721

Question 102 | AM0722

Which technique improves performance by dynamically reordering instructions to maximize instruction-level parallelism?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Out-of-order execution

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Out-of-order execution.

COA36 describes out-of-order execution as scheduling ready operations before stalled older ones while preserving architectural results and exceptions. This is dynamic reordering to exploit instruction-level parallelism. Superscalar issue can provide execution width but does not itself define such reordering. Pipelining overlaps stages and caching improves reuse; neither names the readiness-based dynamic scheduling mechanism.

Why each choice is right or wrong

A - Incorrect: Pipelining overlaps stages; dynamic instruction reordering requires an additional readiness-based execution mechanism.

B - Incorrect: Caching keeps reusable memory copies and does not itself reorder instruction execution.

C - Incorrect: Superscalar issue permits multiple instructions per cycle but does not necessarily require out-of-order scheduling.

D - Correct: Out-of-order execution schedules ready instructions around stalled older ones while preserving required results.

Primary note proof

Study-note section COA36

> Out-of-order execution schedules ready operations before stalled older ones while preserving architectural results and exception behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 114 | lines 2-10

> The following are important points about parallel processing:
> • A parallel processing system is able to perform concurrent data processing to achieve faster execution time.

• The system may have two or more ALUs and be able to execute two or more instructions at the same time
• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 722

Question 103 | AM0723

Which technique improves performance by dynamically allocating processor resources based on program behavior?

A. Pipelining

B. Caching

C. Superscalar architecture

D. Dynamic resource allocation

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer D: Dynamic resource allocation.

COA36 explicitly defines dynamic resource allocation as assigning available internal entries/units at runtime. That definition directly fits the stem's allocation based on behavior during execution. Pipelining describes overlapping stages, caching describes memory reuse, and superscalar execution describes issuing several instructions. They may use allocated resources, but none is the specific allocation technique named by the question.

Why each choice is right or wrong

A - Incorrect: Pipelining defines overlapping instruction stages rather than the general assignment of runtime internal resources.

B - Incorrect: Caching retains memory copies; it does not name the runtime assignment of processor entries and units.

C - Incorrect: Superscalar execution names multiple issue, rather than the allocation process used to assign available resources.

D - Correct: The note explicitly defines this as assigning available internal processor entries and units at runtime.

Primary note proof

Study-note section COA36

> Dynamic resource allocation assigns available internal entries/units at runtime.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 114 | lines 2-10

> The following are important points about parallel processing:
> • A parallel processing system is able to perform concurrent data processing to achieve faster execution time.
> • The system may have two or more ALUs and be able to execute two or more instructions at the same time

• Also, the system may have two or more processors operating concurrently
• Goal is to increase the throughput - the amount of processing that can be accomplished during a given interval of time
Parallel processing can be classified from:

Computer Organization and Architecture module | PDF page 59 | lines 2-15

The control unit that operates the CPU bus system directs the information flow through the register and ALU by selecting the various components in the system. For example, to perform the operation
R1 R2 + R3, the control must provide binary selection variables to the following selector inputs:
1. MUX A selector (SELA): to place the content of R2 into bus A.
2. MUX selector (SELB): to place the content of R3 into bus B.
3. ALU operation selector (OPR): to provide the arithmetic add A +B.
4. Decoder destination selector (SELD): to transfer the content of output bus in to R1.
The four control selection variables are generated in the control unit and must be available at the beginning of a clock cycle. The data from the two source registers propagates through the gates in the multiplexers and the ALU, to the output bus, and into the inputs of the destination register, all during the clock cycle interval. Then, when the next clock transition occurs, the binary information from the output bus is transferred into R1. To achieve a fast response

Computer Organization and Architecture module | PDF page 116 | lines 2-6

8.3 Pipelining
Pipelining is an implementation technique where multiple instructions are overlapped in execution. Pipelining refers to the technique in which a given task is divided into a number of subtasks that need to be performed in sequence. Each subtask is performed by a given functional unit

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 723

Question 104 | AM0724

Which listed item is a fundamental logic building block from which computer hardware is constructed?

A. CPU

B. Memory

C. Hard disk

D. Logic gate

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Logic gate

Logic gates perform elementary Boolean functions and are fundamental building blocks of digital hardware. A CPU, memory system and hard disk are larger components built from many lower-level mechanisms. The final option identifies the logic-level building block. The question explicitly states the level of abstraction so that basic does not mean an arbitrary major system component.

Why each choice is right or wrong

A - Incorrect: A CPU is a major assembled component rather than a fundamental logic building block.

B - Incorrect: Memory is a storage component rather than the requested logic-level building block.

C - Incorrect: A hard disk is a complex storage device.

D - Correct: A logic gate is the fundamental logic-level building block requested.

Primary note proof

Study-note section COA01

> Gates are fundamental logic building blocks; distinguish that level from CPU, memory, disk and motherboard components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 7 | lines 3-9

> Binary information in digital computers represented by physical quantities called signal.
> The fundamental building block of all digital logic circuits is the gate. Logical functions are implemented by the interconnection of gates. A gate is an electronic circuit that produces an

output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. Note that the inversion (NOT) operation is indicated by a circle.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Question sources

Computer Science MCQ practice collection · Question 724

Question 105 | AM0725

Which component is responsible for executing instructions and performing calculations in a computer?

A. CPU

B. Memory

C. Hard disk

D. Motherboard

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: CPU

The CPU contains the control unit, ALU and execution registers. Control sequences operations and transfers, while the ALU performs arithmetic and logic, supporting the CPU's instruction-processing role. Main memory supplies addressed working storage, a hard disk supplies persistent storage, and the motherboard connects components. Those roles distinguish the processing component from the storage and interconnection choices.

Why each choice is right or wrong

A - Correct: CPU's processing/execution function is explicit (pp.6,57).

B - Incorrect: Main memory stores data (p.6), rather than being the arithmetic execution unit.

C - Incorrect: Disk is secondary storage (p.94), not the processing unit.

D - Incorrect: The motherboard physically interconnects components; the CPU carries out the instruction execution and calculations.

Primary note proof

Study-note section COA01

> The CPU contains an ALU, control unit and registers; a microcontroller is a complete integrated control system, not another mandatory CPU subunit.

Study-note section COA11

> The ALU computes; the control unit sequences and enables the transfers.

Study-note section COA01

> A motherboard physically connects components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers

and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Computer Science MCQ practice collection · Question 725

Question 106 | AM0726

Which component retains stored data and instructions without power? Here Memory means ordinary volatile main RAM, and CPU excludes separate firmware storage.

A. CPU

B. Memory

C. Hard disk

D. Motherboard

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Hard disk

An HDD preserves magnetic patterns without continuous power, so the hard disk retains the stored data and instructions. The question now restricts Memory to ordinary volatile main RAM, avoiding the broader meaning that would include ROM or flash. CPU and motherboard name processing and interconnection components rather than the persistent storage mechanism sought. These scope conditions make the intended retention comparison definite.

Why each choice is right or wrong

A - Incorrect: The CPU is not the specified persistent storage component.

B - Incorrect: Ordinary main RAM is volatile under the stated meaning.

C - Correct: An HDD retains its stored bits without continuous power.

D - Incorrect: The motherboard interconnects components rather than naming the persistent storage mechanism.

Primary note proof

Study-note section COA28

> It does not need continuous power to preserve stored bits.

Study-note section COA24

> Both are ordinarily volatile.

Study-note section COA05

> Ordinary flip-flops and SRAM lose state when power is removed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 91 | lines 8-15

> 6.4 Secondary memory
> Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
> • It stores information that is not necessarily in current use.
> • It is slower and having higher capacity than primary memory.
> • This kind of memory is large, slow and inexpensive.
> • It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
> • Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

> 6.2 Main Memory
> The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

> DYNAMIC RAM (DRAM)
> is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
> STATIC RAM (SRAM)
> In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major

parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 81 | lines 6-16

Static RAM:
• a bit of data is stored using the state of a flip-flop.
• Applying power is enough (no need for refreshing).
• Retains value indefinitely, as long as it is kept powered.
• Mostly uses to create cache memory of CPU
• Faster and more expensive than DRAM.
ROM
ROM is used for storing programs that are permanently resident in the computer and for tables of constants that do not change in value once the production of the computer is completed The ROM portion of main memory is needed for storing an initial program called bootstrap loader, which is to start the computer software operating when power is

Question sources

Computer Science MCQ practice collection · Question 726

Question 107 | AM0727

The motherboard of a computer is responsible for:

A. Executing instructions

B. Storing data

C. Connecting different components

D. Performing calculations

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: Connecting different components.

COA01 explicitly states that a motherboard physically connects components. The same section locates the ALU, control unit and registers in the CPU; computation and instruction execution therefore belong to processing components rather than the board's defining role. Memory provides data storage. The question asks the motherboard's responsibility, for which physical connection is the specific supported answer.

Why each choice is right or wrong

A - Incorrect: Instructions are executed by the CPU; that is distinct from the motherboard's physical connection role.

B - Incorrect: Data storage belongs to memory/storage components, rather than defining what the motherboard itself does.

C - Correct: The note explicitly identifies the motherboard as physically connecting the computer's components.

D - Incorrect: Calculations belong to CPU arithmetic/logic hardware rather than the motherboard's defining connection function.

Primary note proof

Study-note section COA01

> A motherboard physically connects components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

> There are four main structural components:
> • Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
> • Main memory: Stores data.
> • I/O: Moves data between the computer and its external environment.
> • System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes

the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Computer Science MCQ practice collection · Question 727

Question 108 | AM0728

Which component supplies the addressable main working store for a running program, distinct from individual CPU execution registers and persistent disk storage?

A. CPU

B. Main memory (RAM)

C. Hard disk

D. Motherboard

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Main memory (RAM)

Main RAM holds the addressed working instructions and data used by a running program. CPU registers hold particular fetched instructions, operands or results for immediate execution, and disks retain information persistently. The motherboard connects these components. Specifying the main working-store role prevents the broad temporary-storage description from also fitting CPU registers or cache.

Why each choice is right or wrong

A - Incorrect: The CPU executes operations and has immediate registers, but is not the main addressed RAM store requested.

B - Correct: Main RAM is the addressable working store for the running program’s instructions and data.

C - Incorrect: A hard disk supplies persistent storage rather than the main RAM working store.

D - Incorrect: The motherboard provides physical component connections rather than this storage role.

Primary note proof

Study-note section COA02

In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA24

DRAM stores charge and requires refresh: denser and commonly used for main memory.

Study-note section COA14

Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Study-note section COA28

It does not need continuous power to preserve stored bits.

Study-note section COA01

A motherboard physically connects components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

6.2 Main Memory
The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 80 | lines 5-11

DYNAMIC RAM (DRAM)
is made with cells that store data as charge on capacitors. The presence or absence of charge in a capacitor is interpreted as a binary 1 or 0. Because capacitors have a natural tendency to discharge, DRAMs require periodic charge refreshing to maintain data storage.
STATIC RAM (SRAM)
In a SRAM, binary values are stored using traditional flip-flop logic-gate. A static RAM will hold its data as long as power is supplied to it. Static RAM's are faster than dynamic RAM's.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)

• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.
• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Computer Science MCQ practice collection · Question 728

Question 109 | AM0729

Which type of software controls and manages the hardware resources of a computer?

A. Operating system

B. Application software

C. Programming language

D. Compiler

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Operating system.

COA02 directly defines the operating system as managing resources and providing services. Applications solve user tasks, and compilers translate source into target code. A programming language is a means of expressing programs rather than the software resource manager. Therefore the operating system is the single option whose stated role matches controlling and managing computer hardware resources.

Why each choice is right or wrong

A - Correct: The operating system is explicitly defined as the resource manager and provider of system services.

B - Incorrect: Applications solve user tasks using system services rather than serving as the general hardware resource manager.

C - Incorrect: A programming language expresses programs; it is not itself the system software managing hardware resources.

D - Incorrect: A compiler translates source code into target code rather than performing general OS resource management.

Primary note proof

Study-note section COA02

An operating system manages resources and provides services.

Study-note section COA02

Applications solve user tasks.

Study-note section COA02

A compiler translates source into target code; assembly is a symbolic form of machine instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 91 | lines 2-7

It must be mapped onto an actual (physical) main memory address by the operating system using mapper.
If the page is present in the main memory, CPU gets the required data from the main memory.
If the mapper detects that the requested page is not present in main memory, a page fault occurs and the page must be read from secondary storage into a page frame in main memory.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline

• Compiler support for efficient transmission high-level language programs into machine language programs

Question sources

Computer Science MCQ practice collection · Question 729

Question 110 | AM0730

Which software allows users to perform specific tasks or functions on a computer?

A. Operating system

B. Application software

C. Programming language

D. Compiler

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: Application software.

COA02 states that applications solve user tasks. This directly matches software allowing users to perform particular tasks or functions. The operating system instead manages resources and provides services; a compiler translates programs, and a programming language expresses them. These support applications but do not name the general category of user-task software requested here.

Why each choice is right or wrong

A - Incorrect: The OS supplies resource management and services rather than naming the category of specific user-task programs.

B - Correct: Applications are explicitly described as software solving user tasks, matching the question’s requested category.

C - Incorrect: A programming language is a notation for programs, not the category of software performing user tasks.

D - Incorrect: A compiler specifically translates code; application software is the general user-task category requested.

Primary note proof

Study-note section COA02

An operating system manages resources and provides services.

Study-note section COA02

Applications solve user tasks.

Study-note section COA02

A compiler translates source into target code; assembly is a symbolic form of machine instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Question sources

Computer Science MCQ practice collection · Question 730

Question 111 | AM0731

Which component provides main working storage for the program’s instructions and data, rather than the CPU’s immediate execution registers or persistent disk storage?

A. CPU

B. Main memory (RAM)

C. Hard disk

D. Motherboard

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Main memory (RAM)

Main memory supplies the addressed working storage that holds a program's instructions and data during operation. CPU registers retain particular fetched instructions, operands and results for immediate execution, while a hard disk supplies persistent storage. The revised stem and expanded memory label distinguish these levels rather than assuming all temporary storage belongs exclusively to one component. A motherboard provides physical interconnection rather than the requested storage role.

Why each choice is right or wrong

A - Incorrect: CPU registers serve immediate execution rather than the main working-storage level requested.

B - Correct: Main RAM is the addressed working store for program instructions and data.

C - Incorrect: A hard disk provides persistent backing storage rather than main working storage.

D - Incorrect: The motherboard physically connects components.

Primary note proof

Study-note section COA02

> In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA24

> DRAM stores charge and requires refresh: denser and commonly used for main memory.

Study-note section COA14

> Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Study-note section COA28

> It does not need continuous power to preserve stored bits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 14-16

> 6.2 Main Memory
> The main memory (RAM) stores data and instructions. the capacity of the memory is 128 words of 8 bits (one byte) per word.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 91 | lines 8-15

6.4 Secondary memory
Devises that provides backup storage are called an Auxiliary memory (Secondary memory)
• It stores information that is not necessarily in current use.
• It is slower and having higher capacity than primary memory.

• This kind of memory is large, slow and inexpensive.
• It is non-volatile storage media i.e. the contents are not erased when the power is switched off.
• Magnetic disk, Magnetic tape and optical disk are the examples of secondary storage.

Question sources

Computer Science MCQ practice collection · Question 731

Question 112 | AM0732

Which type of software translates high-level programming languages into machine code?

A. Operating system

B. Application software

C. Programming language

D. Compiler

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: Compiler

A compiler translates source code into target code; in this question the source is a high-level language and the target is machine code. The operating system manages resources, applications solve user tasks, and a programming language is the notation in which a program is expressed. The requested translation function therefore belongs to the compiler category.

Why each choice is right or wrong

A - Incorrect: An operating system manages resources and provides services; compiler is the translation category requested.

B - Incorrect: Application software solves user tasks rather than naming the general source-to-target translation category.

C - Incorrect: The passage distinguishes source high-level language from resulting machine language; a language itself is not the named translating agent (p.77).

D - Correct: Compiler support is explicitly tied to high-level-to-machine translation (p.77).

Primary note proof

Study-note section COA02

An operating system manages resources and provides services.

Study-note section COA02

Applications solve user tasks.

Study-note section COA02

A compiler translates source into target code; assembly is a symbolic form of machine instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Question sources

Computer Science MCQ practice collection · Question 732

Question 113 | AM0733

The system clock in a computer is responsible for:

A. Controlling the execution speed of instructions

B. Storing data and instructions

C. Managing the input/output devices

D. Connecting different hardware components

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer A: Controlling the execution speed of instructions

Control selections are available at the beginning of a clock cycle and register data are captured on the next clock transition (PDF p.59). Fetch timing likewise advances after each clock pulse (p.55). Thus A is the closest description of clock timing instruction steps; it does not mean frequency alone determines total execution time.

Why each choice is right or wrong

A - Correct: Clock transitions time the operations and sequence steps (pp.55,59); this regulates execution pace.

B - Incorrect: Registers and memory store bits; the clock times their updates rather than stores the program (p.59).

C - Incorrect: I/O management is assigned to interfaces/control mechanisms, not solely the clock (pp.101-102).

D - Incorrect: Interconnection provides communication (p.6); the clock provides timing rather than that physical connection.

Primary note proof

Study-note section COA05

> Clocked sequential circuits update at designated clock events; combinational logic does not inherently require a clock.

Study-note section COA34

> CPU time=instruction_count×CPI×clock_period=instruction_count×CPI/clock_rate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 59 | lines 2-15

> The control unit that operates the CPU bus system directs the information flow through the register and ALU by selecting the various components in the system. For example, to perform the operation
> R1 R2 + R3, the control must provide binary selection variables to the following selector inputs:
> 1. MUX A selector (SELA): to place the content of R2 into bus A.
> 2. MUX selector (SELB): to place the content of R3 into bus B.
> 3. ALU operation selector (OPR): to provide the arithmetic add A +B.
> 4. Decoder destination selector (SELD): to transfer the content of output bus in to R1.
> The four control selection variables are generated in the control unit and must be available at the beginning of a clock cycle. The data from the two source registers propagates through the gates in the multiplexers and the ALU, to the output bus, and into the inputs of the destination register, all during the clock cycle interval. Then, when the next clock transition occurs, the binary information from the output bus is transferred into R1. To achieve a fast response

Computer Organization and Architecture module | PDF page 55 | lines 9-28

> I. Fetch the instruction
> Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be

brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Computer Organization and Architecture module | PDF page 6 | lines 2-8

There are four main structural components:
• Central processing unit (CPU): Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
• Main memory: Stores data.
• I/O: Moves data between the computer and its external environment.
• System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

Computer Organization and Architecture module | PDF page 101 | lines 10-16

7.4 Input/output Problems
Wide variety of peripherals and delivering different amounts of data per second Work at different speeds Send/receive data in different formats all slower than CPU and RAM. Hence I/O modules are used as a solution.
Input/output Module: It is the entity within a computer that is responsible for the control of one or more external devices and for the exchange of data between those devices and main memory and/or CPU.

Question sources

Computer Science MCQ practice collection · Question 733

Question 114 | AM0734

Which process is responsible for fetching instructions from memory?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Fetching

The instruction-pipeline list explicitly begins “Fetch the instruction from memory” (PDF p.117), and the fetch micro-operations copy memory into IR (p.55). The question names this retrieval phase, not interpretation or computation.

Why each choice is right or wrong

A - Incorrect: Decoding interprets the retrieved instruction after fetching (p.117).

B - Incorrect: Execution carries out the operation later in the pipeline (p.117).

C - Correct: Fetching is the memory-to-instruction-register retrieval phase (pp.55,117).

D - Incorrect: Storing writes the result after execution, the opposite transfer purpose (p.117).

Primary note proof

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

> Instruction Pipeline:
> Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
> 1. Fetch the instruction from memory
> 2. Decode the instruction
> 3. Calculate the effective address
> 4. Fetch the operands from memory
> 5. Execute the instruction
> 6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 55 | lines 9-28

> I. Fetch the instruction
> Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock

pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Computer Organization and Architecture module | PDF page 70 | lines 17-24

The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

Computer Science MCQ practice collection · Question 734

Question 115 | AM0735

Which process decodes the fetched instruction and determines the necessary actions?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Decoding

The module says decoding determines the operation, addressing mode and operand location (PDF p.70). It follows fetch and precedes execution in the instruction phases (p.117).

Why each choice is right or wrong

A - Correct: Decoding explicitly determines the required operation and addressing details.

B - Incorrect: Execution performs the operation after it has been determined (p.117).

C - Incorrect: Fetching retrieves the instruction bits before they are interpreted (p.117).

D - Incorrect: Storing writes the computed result, not the operation interpretation (p.117).

Primary note proof

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

> Instruction Pipeline:
> Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
> 1. Fetch the instruction from memory
> 2. Decode the instruction
> 3. Calculate the effective address
> 4. Fetch the operands from memory

5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 55 | lines 2-8

1. Fetch an instruction from memory
2. Decode the instruction
3. Read the effective address from memory if the instruction has an indirect address
4. Execute the instruction
Upon the completion of step 4, the control goes back to step 1 to fetch, decode and execute the next instruction. This process continues indefinitely unless a HALT instruction is encountered.

Question sources

Computer Science MCQ practice collection · Question 735

Question 116 | AM0736

Which process carries out the actual operations or calculations specified by the instruction?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Execution

The instruction stages separate fetch, decode, operand access, execution and result storage (PDF p.117). ALU supplies the required micro-operations for execution (p.57), so actual calculation is execution rather than any transport phase.

Why each choice is right or wrong

A - Incorrect: Decoding determines what to do, not the arithmetic itself.

B - Correct: Execution is the listed phase that performs the operation (pp.57,117).

C - Incorrect: Fetching retrieves the instruction and precedes the calculation (p.117).

D - Incorrect: Storing preserves the result after calculation (p.117).

Primary note proof

Study-note section COA15

A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA15

The control unit coordinates fetching and data flow; the ALU performs arithmetic and logic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division.

> Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

Computer Science MCQ practice collection · Question 736

Question 117 | AM0737

Which process is responsible for storing the results of the executed instruction?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Storing

The instruction-pipeline sequence ends by storing the result in its proper place (PDF p.117). The RISC variant specifies writing the result into a register (p.118), establishing the result-preservation role of D.

Why each choice is right or wrong

A - Incorrect: Decoding interprets operation and addressing fields, before a result exists.

B - Incorrect: Execution produces a result; the listed store phase preserves it afterward (p.117).

C - Incorrect: Fetching brings in the instruction, not its eventual result (p.117).

D - Correct: Store result/write register is the explicit final stage (pp.117-118).

Primary note proof

Study-note section COA15

A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Study-note section COA15

A write-back/result-storage phase is not another fetch.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 118 | lines 3-19

A RISC (Reduced Instruction Set Computer) processor pipeline operates in much the same way, although the stages in the pipeline are different. While different processors have different numbers of steps, they have basically variations of these five steps:
1. fetch instructions from memory
2. read registers and decode the instruction
3. execute the instruction or calculate an address
4. access an operand in data memory
5. write the result into a register
The length of the pipeline is dependent on the length of the longest step. Because RISC instructions are simpler than those used in pre-RISC processors (now called CISC, or Complex Instruction Set Computer), they are more conducive to pipelining. While CISC instructions varied in length, RISC instructions are all the same length and can be fetched in a single operation. Ideally, each of the stages in a RISC processor pipeline should take 1 clock cycle so that the processor finishes an instruction each clock cycle and averages one cycle per instruction (CPI). Among the characteristics attributed to RISC is its ability to use an efficient instruction pipeline.

Question sources

Computer Science MCQ practice collection · Question 737

Question 118 | AM0738

Which unit of the CPU is responsible for fetching instructions and data from memory?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Control Unit (CU)

The pipeline description explicitly says the control unit initiates the next fetch when there is buffer space (PDF p.117). CPU control supervises register transfers and ALU work (p.57); the fetch micro-operations transfer address and instruction bits (p.55). This is coordinating responsibility, not a claim the CU physically stores all fetched data.

Why each choice is right or wrong

A - Incorrect: ALU performs the arithmetic/logical micro-operations, not the fetch-sequencing role (p.57).

B - Correct: CU initiates fetch and supervises the register transfers (pp.57,117).

C - Incorrect: Cache supplies stored copies on hits (p.84); it does not decide the instruction-cycle control sequence.

D - Incorrect: Registers hold address/data/instruction values, while CU controls their transfers (pp.48,57).

Primary note proof

Study-note section COA15

> The control unit coordinates fetching and data flow; the ALU performs arithmetic and logic.

Study-note section COA15

> A sample fetch is MAR←PC; MDR←M[MAR]; IR←MDR; PC←PC+instruction_length.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 117 | lines 2-14

> Instruction Pipeline:
> Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The

following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 55 | lines 9-28

I. Fetch the instruction
Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Question sources

Computer Science MCQ practice collection · Question 738

Question 119 | AM0739

Which unit of the CPU performs arithmetic and logical operations?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer A: Arithmetic Logic Unit (ALU)

ALU is explicitly the CPU part that performs required micro-operations; the module classifies arithmetic and logical manipulation as computational operations (PDF pp.57,74-75). Control selects the operation and registers supply/receive values.

Why each choice is right or wrong

A - Correct: ALU performs required arithmetic/logical micro-operations (p.57).

B - Incorrect: CU directs transfers and tells ALU which operation to perform (p.57).

C - Incorrect: Cache is a fast storage buffer (p.84), not the computation unit.

D - Incorrect: Register set stores intermediate data; ALU operates on it (p.57).

Primary note proof

Study-note section COA11

> The ALU computes; the control unit sequences and enables the transfers.

Study-note section COA01

> The CPU contains an ALU, control unit and registers; a microcontroller is a complete integrated control system, not another mandatory CPU subunit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU

performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 739

Question 120 | AM0740

Which CPU storage class holds explicit execution operands and results, rather than copies of memory blocks?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Register Unit

Registers hold explicit execution operands and results. Cache also stores temporary information, but its role is retaining copies of memory blocks near the processor. The revised question uses that role distinction, so temporary storage alone no longer makes both choices correct. The ALU performs arithmetic and logic, and the control unit coordinates operations rather than being the requested operand-storage class.

Why each choice is right or wrong

A - Incorrect: The ALU computes rather than being the operand-storage class.

B - Incorrect: The control unit sequences operations rather than holding explicit execution operands as its defining role.

C - Incorrect: Cache holds copies of memory blocks.

D - Correct: Registers hold explicit execution operands and results.

Primary note proof

Study-note section COA14

> Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Study-note section COA14

> The accumulator holds operands/results, SP tracks the stack, and status flags report conditions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions.

The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

6.3 Cache Memory
Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
• The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
• When CPU needs to access memory, the cache is examined
• If the word is found in the cache, it is read from the fast memory
• If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
• When the CPU refers to memory and finds the word in cache, it is said to produce a hit
• Otherwise, it is a miss

Question sources

Computer Science MCQ practice collection · Question 740

Question 121 | AM0741

Which component of the CPU controls the flow of data and instructions within the CPU?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Control Unit (CU)

CU explicitly supervises information transfers among registers and tells ALU what operation to perform (PDF p.57). The bus-control example selects sources, operation and destination (p.59), matching control of internal flow.

Why each choice is right or wrong

A - Incorrect: ALU processes operands selected by control (pp.57,59), rather than directs the overall flow.

B - Correct: CU selects register sources, destination and ALU operation (p.59).

C - Incorrect: Cache stores instruction/data copies (p.84), not the CPU transfer-control sequence.

D - Incorrect: Registers hold intermediate data; CU controls their transfers (p.57).

Primary note proof

Study-note section COA11

> The ALU computes; the control unit sequences and enables the transfers.

Study-note section COA14

> A control word chooses ALU sources, destination and operation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

> 5.11. Central Processing Unit.
> The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the

computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Computer Organization and Architecture module | PDF page 59 | lines 2-15

The control unit that operates the CPU bus system directs the information flow through the register and ALU by selecting the various components in the system. For example, to perform the operation
R1 R2 + R3, the control must provide binary selection variables to the following selector inputs:
1. MUX A selector (SELA): to place the content of R2 into bus A.
2. MUX selector (SELB): to place the content of R3 into bus B.
3. ALU operation selector (OPR): to provide the arithmetic add A +B.
4. Decoder destination selector (SELD): to transfer the content of output bus in to R1.
The four control selection variables are generated in the control unit and must be available at the beginning of a clock cycle. The data from the two source registers propagates through the gates in the multiplexers and the ALU, to the output bus, and into the inputs of the destination register, all during the clock cycle interval. Then, when the next clock transition occurs, the binary information from the output bus is transferred into R1. To achieve a fast response

Question sources

Computer Science MCQ practice collection · Question 741

Question 122 | AM0742

Which CPU-near storage component retains copies of memory blocks to speed repeated or nearby memory accesses?

A. Arithmetic Logic Unit (ALU)

B. Control Unit (CU)

C. Cache Unit

D. Register Unit

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Cache Unit

Cache retains copies of memory blocks near the CPU and exploits temporal and spatial locality. Registers instead hold explicit execution operands and results, even though frequently used values may also remain there. The ALU computes values and the control unit sequences operations. The memory-block-copy criterion therefore identifies cache without claiming that no other CPU component can hold useful data.

Why each choice is right or wrong

A - Incorrect: The ALU computes arithmetic and logic rather than retaining a memory-block-copy tier.

B - Incorrect: The control unit sequences transfers and operations rather than supplying this data-copy storage tier.

C - Correct: Cache stores nearby copies of memory blocks for repeated or spatially local accesses.

D - Incorrect: Registers hold explicit execution operands/results; that differs from the memory-block-copy cache role.

Primary note proof

Study-note section COA23

> Cache copies memory blocks near the CPU.

Study-note section COA23

> Temporal locality reuses recent data; spatial locality uses nearby locations.

Study-note section COA14

> Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Study-note section COA11

> The ALU computes; the control unit sequences and enables the transfers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 84 | lines 2-16

> 6.3 Cache Memory
> Cache memory is a small, high-speed RAM buffer located between the CPU and main memory.
> Cache memory holds a copy of the instructions (instruction cache) or data (operand or data cache) currently being used by the CPU.
> The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.
> • The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
> • When CPU needs to access memory, the cache is examined
> • If the word is found in the cache, it is read from the fast memory
> • If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word
> • When the CPU refers to memory and finds the word in cache, it is said to produce a hit
> • Otherwise, it is a miss

Computer Organization and Architecture module | PDF page 85 | lines 2-8

• The performance of cache memory is frequently measured in terms of a quantity called hit ratio
Hit ratio = hit / (hit+miss)
• The basic characteristic of cache memory is its fast access time
• Therefore, very little or no time must be wasted when searching the words in the cache
• The transformation of data from main memory to cache memory is referred to as a mapping process.

Computer Organization and Architecture module | PDF page 57 | lines 10-21

5.11. Central Processing Unit.
The part of the computer that performs the bulk of data processing operations is called the Central processing unit and is referred to as the CPU. The CPU is made up of three major parts The register set stores intermediate data used during the execution of the instructions. The Arithmetic and Logic unit (ALU) performs the required micro-operations for executing the instructions. The control unit supervises the transfer of information among the registers and instructs the arithmetic and logic units as to which operation to perform. The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer. Computer architecture is sometimes defined as the computer structure and behavior as seen by the programmer that uses machine language instructions. This includes the instruction formats, addressing modes, the instruction sets and the general organizations of the CPU registers

Question sources

Computer Science MCQ practice collection · Question 742

Question 123 | AM0743

During a taken branch instruction, which phase computes or applies the branch target to replace the sequential next-PC value?

A. Decoding

B. Execution

C. Fetching

D. Storing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Execution

A fetch uses the selected PC to obtain an instruction and can advance it to the sequential address. Executing a taken branch can later replace that sequential value with the branch target. The question now specifically asks about that branch action, rather than the broader process of selecting, reading or advancing a next instruction address. Decode interprets fields and write-back stores results; neither is the conceptual branch-execution role requested.

Why each choice is right or wrong

A - Incorrect: Decoding interprets opcode and addressing fields rather than performing the specified branch action.

B - Correct: Executing the taken branch replaces the sequential PC with its target.

C - Incorrect: Fetching obtains the instruction at the selected address rather than performing this branch action.

D - Incorrect: Result storage is not the conceptual branch-target execution phase.

Primary note proof

Study-note section COA15

> A branch can replace the sequential PC later.

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

> I. Fetch the instruction
> Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
> T0: AR ← PC
> Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
> T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1, INR(PC) = 1
> II. Decode the instruction
> T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Computer Organization and Architecture module | PDF page 117 | lines 2-14

Instruction Pipeline:
Pipeline processing can occur not only in the data stream but also in the instruction stream as well. An instruction pipeline reads consecutive instructions from memory while previous instructions are being executed in other segments. Whenever there is space in the buffer, the control unit initiates the next instruction fetch phase. The following steps are needed to process each instruction:
1. Fetch the instruction from memory
2. Decode the instruction
3. Calculate the effective address
4. Fetch the operands from memory
5. Execute the instruction
6. Store the result in the proper place

Computer Organization and Architecture module | PDF page 72 | lines 26-32

Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Question sources

Computer Science MCQ practice collection · Question 743

Question 124 | CS0232

What is the purpose of an opcode in a 16-bit instruction format (4-bit opcode | 6-bit operand ref | 6-bit operand ref)?

A. It holds the destination address of an operand

B. It determines the operation to be done

C. It holds the source address of an operand

D. It holds the next instruction reference

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: It determines the operation to be done

The stated field widths sum to 16 bits: 4+6+6. Their widths do not change the opcode's purpose: operation selection (PDF p.47). Operand reference fields identify source/destination values or locations; PC tracks sequence.

Why each choice is right or wrong

A - Incorrect: Destination reference is an operand/address field, not the opcode (p.47).

B - Correct: Operation selection is the explicit opcode definition (p.47).

C - Incorrect: Source reference belongs to the operand/address fields (p.47).

D - Incorrect: Next instruction is tracked by PC (p.70), not the opcode.

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Study-note section COA16

> A 4-bit opcode permits at most 16 encodings, not necessarily 16 implemented instructions.

Study-note section COA16

> A 6-bit operand-reference field has 64 encodings; what they designate depends on the machine.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 11-18

> An Instruction Code is a group of bits that instructs the computer to perform a specific operation. It is usually divided into parts, each having its own particular interpretation. The most basic part of an instruction code is its operation part. The operation code of an instruction is a group of bits that defines such operations as Add, Subtract, Multiply, Shift and Complement. The operation part of an instruction code specifies the operation to be performed. This operation must be performed on some data stored in processor registers or in memory. An instruction must specify not only the operation but also the registers or the memory words where the operands are to be found.

Computer Organization and Architecture module | PDF page 70 | lines 17-24

> The control unit of a computer is designed to go through an instruction cycle that is divided into three major phases. 1. Fetch the instruction from memory 2. Decode the instruction 3. Execute the instruction
> There is one register in the computer called the program counter (PC) that keeps track of the

instructions in the program stored in memory. PC holds the address of the instruction to be executed next and incremented each time an instruction is fetched from memory. The decoding done in step 2, determines the operation to be performed, the addressing mode of the instruction, and the location of the operand.

Question sources

Computer Science Exit Exam practice collection · Question 232

Question 125 | CS0234

Which performance metric can be used to measure the time taken to perform a read or write operation?

A. Transfer rate

B. Memory cycle time

C. Fetching time

D. Access time

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: Access time

Access time is the elapsed time from initiation to completion of a memory operation (PDF p.83). Memory cycle time is the minimum separation of successive operations. This distinguishes one read/write duration from transfer throughput or instruction fetching.

Why each choice is right or wrong

A - Incorrect: Transfer rate measures an amount per unit time; it is not the single-operation interval defined on p.83.

B - Incorrect: Cycle time measures spacing of successive operations, not the requested initiation-to-completion duration (p.83).

C - Incorrect: Fetching is one instruction-cycle task, not the general read/write performance metric (p.55).

D - Correct: Access time is exactly the operation-duration definition (p.83).

Primary note proof

Study-note section COA23

Access time measures latency of a read/write; cycle time is the minimum interval before another operation and may include recovery.

Study-note section COA23

Transfer rate measures data per time, not individual-request delay.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 83 | lines 11-14

The time that elapses between the initiation of an operation and completion of that operation is called memory access time. The minimum time delay between two successive memory operations is called memory cycle time. The cycle time is usually slightly lower than the access time.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

6.1 Memory Hierarchy:
A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
Figure 6.1 memory hierarchy
• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 55 | lines 9-28

I. Fetch the instruction
Initially, the program counter PC is loaded with the address of the first instruction in the program and SC is cleared to 0, providing a decoded timing signal T0. After each clock pulse, SC is incremented by one, so that the timing signals go through a sequence T0, T1, T2 and so on. Since the address lines of the memory unit are hardwired to AR, address of the instruction to be fetched must first be placed in AR. Thus the first register transfer in an instruction cycle should be transferring the content of PC (address of the instruction to be brought in for execution) to AR
T0: AR ← PC
Bus selects PC, LD(AR) = 1. Next the current instruction is brought from memory into IR and PC is incremented
T1: IR ← M[AR], PC ← PC + 1. Bus selects RAM, Memory read is set, LD(IR) = 1,
INR(PC) = 1
II. Decode the instruction
T2: D0, ... , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)
Therefore, micro-operations for the fetch and decode phases can be specified by the following register transfer statements.
T0: AR ← PC
T1: IR ← M[AR], PC ← PC + 1
T2: D0, ..., D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Question sources

Computer Science Exit Exam practice collection · Question 234

Question 126 | CS0235

Under the textbook access-method taxonomy, how is magnetic-disk block access classified when requests identify blocks but mechanical positioning introduces delay?

A. Associative

B. Sequential

C. Random

D. Direct

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer D: Direct

The stated textbook taxonomy calls magnetic disk access direct: a request identifies an addressed block, but positioning creates access delay. RAM is classified as random access with broadly address-independent timing, while tape is sequential and associative access matches content. Informal usage may call disks random-access devices too, but the revised question explicitly chooses the textbook classification convention rather than treating the labels as universally exclusive.

Why each choice is right or wrong

A - Incorrect: Associative access matches by content rather than addressed disk blocks.

B - Incorrect: Tape-like sequential access is not the disk's required access pattern.

C - Incorrect: Random is the RAM category in the explicitly stated textbook taxonomy.

D - Correct: Direct is the disk category in that taxonomy.

Primary note proof

Study-note section COA28

> In textbook access taxonomy, tape is sequential, disk is direct access with positioning delay, RAM is random access with broadly address-independent timing, and associative access matches content.

Study-note section COA28

> "Random-access disk" is also common informal usage; specify the taxonomy when comparing direct and random access.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 92 | lines 2-10

6.4.1 Magnetic Disk
• Magnetic disks are the foundation of external memory on virtually all computer systems.
• A disk is a circular platter constructed of nonmagnetic material called the substrate coated with a magnetisable material.
• More suitable than magnetic tapes for a wider range of applications such as supporting direct access of data
• It is a thin, circular plate made of metal & plastic, which is coated with iron-oxide.
• We can randomly access the data.

Computer Organization and Architecture module | PDF page 98 | lines 7-16

6.4.2 Magnetic Tape
• Magnetic Tape is a plastic ribbon which is usually ½ inch or ¼ inch wide & 50 to 2400 feet long.
• It is coated with iron-oxide material.
• It is similar to the tape of audio cassettes of tape recorders. Data is stored as binary digits.
• Data is accessed sequentially so searching becomes difficult.
• Tape systems use the same reading and recording techniques as disk systems..
• Magnetic tape was the first kind of secondary memory.It is still widely used as the lowest-cost, slowest-speed member of the memory hierarchy.

Computer Organization and Architecture module | PDF page 87 | lines 2-15

Associative Mapping
• An associative mapping uses an associative memory.
• This memory is being accessed using its contents.
• Associative memory is expensive compared to RAM
• Each line of cache memory will accommodate the address (main memory) and the contents of that address from the main memory.
• That is why this memory is also called Content Addressable Memory (CAM). It allows each block of main memory to be stored in the cache.
A CPU address of 15 bits is places in the argument register and the associative memory us searched for a matching address
• If the address is found, the corresponding 12bits data is read and sent to the CPU
• If not, the main memory is accessed for the word
If the cache is full, an address-data pair must be displaced to make room for a pair that is needed and not presently in the cache

Question sources

Computer Science Exit Exam practice collection · Question 235

Question 127 | CS0236

Which statement reverses the broad integration trend described for computer evolution?

A. Increasing component size

B. Increasing processor speed

C. Increasing integration density

D. Increasing memory size

Hint

Use the stated historical convention or integration trend, rather than assuming every model has identical characteristics.

Answer A: Increasing component size

The note describes a broad trend toward smaller individual components while density, capacity and processor performance increase. Increasing component size reverses that stated trend. The other alternatives describe growth in performance, integration density or memory capacity and are therefore consistent with it. These are historical tendencies, not guarantees that every later product improves every measured property.

Why each choice is right or wrong

A - Correct: The broad trend reduces individual component dimensions, so increasing size reverses it.

B - Incorrect: Rising processor performance is consistent with the stated evolutionary tendency.

C - Incorrect: Increasing integration density is explicitly part of the stated integration trend.

D - Incorrect: Growing storage capacity is consistent with the stated trend even as individual component dimensions shrink.

Primary note proof

Study-note section COA38

> Integration generally reduced individual component dimensions while capacity, density and processor performance grew; these are trends rather than guarantees for every model.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 79 | lines 3-13

> 6.1 Memory Hierarchy:
> A computer system is equipped with a hierarchy of memory subsystems. There are several memory types with very different physical properties. The important characteristics of memory devices are cost per bit, access time, data transfer rate, alterability and compatibility with processor technologies. Figure shows the hierarchy of memory in a typical memory with a trend in access time, amount of storage, and cost per byte.
> Figure 6.1 memory hierarchy

• Design constraints: How much? How fast? How expensive?
• Faster access time, greater cost per bit
• Greater capacity, smaller cost per bit,
• Greater capacity, slower access time.

Computer Organization and Architecture module | PDF page 77 | lines 2-14

RISC characteristics
• Relatively few instructions
• Relatively few addressing modes
• Memory access limited to load and store instructions
• All operations done within the register of the CPU
• Fixed length, easily decoded instruction format
• Single cycle instruction execution
• Hard-wired rather than micro-programmed control
• A relatively large number of registers in the processor
• Use overlapped register windows to speed up procedure call and return
• Efficient instruction pipeline
• Compiler support for efficient transmission high-level language programs into machine language programs

Computer Organization and Architecture module | PDF page 78 | lines 2-9

5.19 CISC Instruction:
CISC refers to Complex Instruction Set Computer
CISC characteristic
• Large number of instructions - typically from 100 to 250 instructions
• Some instructions that perform specialized tasks and are used infrequently
• A large variety of addressing modes - typically from 5 to 20 different modes
• Variable length instruction formats
• Instructions that manipulate operands in memory RISC characteristics

Computer Organization and Architecture module | PDF page 30 | lines 3-19

3.1 Integrated Circuits
• Integrated circuit (IC) is the basic building block of digital circuits.
• An integrated circuit is a small silicon semiconductor crystal, called a chip.
• The various gates are interconnected inside the chip to form the required circuit.
• As the technology of ICs has improved, the number of gates that can be put in a single chip has increased.
✓ Small-scale integration (SSI) devices contain several (usually less than 10) independent gates in a single package.
✓ Medium-scale integration (MSI) devices contain approximately 10 to 200 gates in a single package.
E.g To form decoders, adders, and registers.
✓ Large-scale integration (LSI) devices contain between 200 and a few thousands gates in a single package.
Eg. processors, memory chips, and programmable modules.
✓ Very-large-scale integration (VLSI) devices contain thousands of gates in a single

package.
E.g large memory arrays and complex microcomputer chips.

Question sources

Computer Science Exit Exam practice collection · Question 236

Question 128 | CS0237

Which one of the following is NOT correct about Von Neumann's architecture?

A. Data and instructions are stored separately

B. Modern computers are based on Von Neumann's architecture

C. Instruction execution is sequential

D. Memory content is located by its address

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Separate instruction and data storage contradicts the textbook Von Neumann model.

COA02 defines the textbook Von Neumann model as sharing an addressed memory system between instructions and data, with normally sequential execution unless control flow changes. Thus separate storage is the defining contradiction. The note allows modern machines to retain a unified address-space view while using split caches; ordinary execution and addressed memory remain compatible with that model.

Why each choice is right or wrong

A - Correct: Separate instruction/data memories describe Harvard organization rather than the textbook shared-memory Von Neumann model.

B - Incorrect: Modern systems can retain a unified address-space view, so this broad conventional association is compatible with the note.

C - Incorrect: Normally sequential instruction progression is part of the textbook model, with branches providing explicit control-flow changes.

D - Incorrect: The model explicitly uses addressed memory, so locating content by address is a compatible statement.

Primary note proof

Study-note section COA02

In the textbook Von Neumann model, instructions and data share an addressed memory system; the CPU normally advances sequentially unless control flow changes.

Study-note section COA02

Harvard separates instruction and data memories or paths.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 47 | lines 20-26

The simplest way to organize a computer is to have one processor register and an instruction code format with two parts. The first part of instruction code specifies the operation to be performed and the second specifies an address. The memory address tells the control where to find an operand in memory. This operand is read from memory and used as the data to be operated on together with the data stored in processor register. Instructions are stored in one section of memory and data in another. The operation is performed with memory operands and the content of AC. Figure 5-1 below depicts this type of organization.

Computer Organization and Architecture module | PDF page 48 | lines 14-23

Computer instructions are usually stored in consecutive memory locations and executed sequentially one at a time. The control reads an instruction from specific address in memory and executes it. It then continues by reading the next instruction in sequence and executes it and so on. This type of instruction sequencing needs a counter to calculate the address of the next instruction after execution of the current instruction is completed. It is also necessary to provide a register in a control unit for storing the instruction code after it is read from the memory. The data register (DR) holds the operand read from memory. The accumulator (AC) register is a general purpose processing register. The instructions read from memory are placed in instruction register (IR). The temporary register (TR) is used for holding temporary data during the processing.

Computer Organization and Architecture module | PDF page 51 | lines 14-22

Instruction Set Completeness
A computer should have a set of instructions so that the user can construct machine language program to evaluate any function that is known to be computable. The set of instructions are said to be complete if the computer includes a sufficient number of instructions in each of the following categories:
1. Arithmetic, Logical and Shift Instructions.
2. Instructions for moving information to and from memory and processor registers.
3. Program control instructions together with instructions that check status conditions.
4. Input and Output instructions.

Question sources

Computer Science Exit Exam practice collection · Question 237

Question 129 | CS0238

Which instruction type is used to test the value of a data word or status of a condition?

A. Data processing instruction

B. Test instruction

C. Data movement instruction

D. Branch instruction

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Test instruction.

COA19 states that TEST/CMP examine values and set condition information, often without retaining an arithmetic result. That exactly matches examining a data word or a condition status. Arithmetic/data processing is a broader computing class, data movement copies values, and branches choose control-flow targets. The named test-instruction category is therefore the direct answer among the options.

Why each choice is right or wrong

A - Incorrect: Data processing is a broader computing category; the stem specifically requests the instruction used to test values.

B - Correct: TEST/CMP examine values and establish condition information, directly matching the question's described operation.

C - Incorrect: Data movement copies values between locations rather than testing their contents or condition status.

D - Incorrect: Branches choose control-flow targets, often using previously established conditions rather than performing the requested test.

Primary note proof

Study-note section COA19

> TEST/CMP examine values and set condition information, often without retaining the arithmetic result.

Study-note section COA19

> Branch instructions choose a target, conditionally or unconditionally.

Study-note section COA19

> CALL saves a return point; RETURN restores it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Organization and Architecture module | PDF page 74 | lines 3-14

Data Manipulation
Data manipulation instructions perform operations on data and provide the computational capabilities for the computer. The data manipulation instructions in a typical computer are usually divided into three basic types.
1. Arithmetic instruction
2. Logical and bit manipulation
3. Shift instruction
a. Arithmetic Instruction
The four basic arithmetic operations are addition, subtraction, multiplication and division. Most computers provide instructions for all four operations. Some small computers have only addition and possibly subtraction instructions. The multiplication and division must then be generated by means of software subroutines.

Computer Organization and Architecture module | PDF page 72 | lines 26-32

Direct Addressing Mode: In this mode the effective address is equal to the address part of the instruction. The operand resides in memory and its address is given directly by the address field of the instruction. In a branch type of instruction the address field specifies the actual branch address.
Indirect Addressing Mode: In this mode the address field of the instruction gives the address where the effective address is stored in memory. Control fetches the instruction from memory and uses its address part to access memory again to read the effective address.

Computer Organization and Architecture module | PDF page 73 | lines 19-24

Data transfer instruction.
Data transfer instructions move data from place to place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor register and input or output, and between the processor registers themselves. Table 5-5 gives lists of eight data transfer instructions used in many computers. Accompanying each instruction is mnemonic symbol.

Computer Organization and Architecture module | PDF page 51 | lines 14-22

Instruction Set Completeness
A computer should have a set of instructions so that the user can construct machine language program to evaluate any function that is known to be computable. The set of instructions are said to be complete if the computer includes a sufficient number of instructions in each of the following categories:
1. Arithmetic, Logical and Shift Instructions.
2. Instructions for moving information to and from memory and processor registers.
3. Program control instructions together with instructions that check status conditions.
4. Input and Output instructions.

Question sources

Computer Science Exit Exam practice collection · Question 238

Question 130 | NS0221

What is the purpose of an opcode in a 16-bit instruction format (4-bit opcode | 6-bit operand ref 6-bit operand ref)?

A. It holds the destination address of an operand

B. It determines the operation to be done

C. It holds the source address of an operand

D. It holds the next instruction reference

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: The opcode specifies the operation.

COA16 defines the opcode as specifying the operation and operand fields as locating data/registers. In the given 16-bit format, the 4-bit opcode therefore selects an operation, while the two 6-bit references identify operands under the machine's interpretation. The bit widths are not evidence that the opcode stores a source/destination address or next-PC reference.

Why each choice is right or wrong

A - Incorrect: A destination operand reference belongs in an operand field, not in the operation-selecting opcode.

B - Correct: The opcode determines which operation the processor should perform for this instruction.

C - Incorrect: A source operand reference is an operand-field role rather than the opcode's operation role.

D - Incorrect: The next-instruction reference is a control-flow/PC concern, not the general purpose of this opcode field.

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Question sources

Computer Science Exit Exam practice collection · Question 232

Question 131 | NS0289

A designer adds a PCI expansion card to a processor whose local bus uses a different transfer protocol. The two sides cannot be wired together directly. Which listed hardware category is needed to adapt transfers between the processor bus and the expansion interconnect?

A. SCSI

B. PCI

C. Bus bridge/controller logic

D. Multi bus

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Bus bridge/controller logic

The scenario requires hardware that adapts transfers between unlike bus segments, so bus bridge/controller logic is needed. PCI identifies the expansion interconnect on one side of the connection; merely naming it does not supply the adaptation to the processor bus. SCSI identifies a storage connection and Multibus another bus family. The distinction is between the interconnect being attached and the hardware that enables the attachment.

Why each choice is right or wrong

A - Incorrect: SCSI is a storage connection rather than the adapter between these unlike bus segments.

B - Incorrect: PCI names the expansion interconnect already present in the scenario, not the adapting component.

C - Correct: Bus bridge/controller logic adapts transfers between the unlike processor and expansion bus segments.

D - Incorrect: Multibus names a bus family rather than the component needed to adapt this connection.

Primary note proof

Study-note section COA32

> PCI is a peripheral expansion interconnect, not a disk protocol; connecting unlike bus segments requires suitable bridge/controller logic.

Question sources

Computer Science Exit Exam · 2018 · Question 6

Question 132 | NS0290

Combinational circuit differ from sequential circuit primarily because

A. Combinational circuit cannot be built using logic circuit

B. The output of combinational circuit depends only on the present input, not on past output.

C. Sequential circuit have no feedback while combinational circuit do

D. The combinational circuit requires clock signal, but sequential circuit do not

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer B: Combinational output depends on present inputs rather than stored history.

COA05 defines combinational output through present inputs after propagation delay, while sequential output also depends on stored state. B expresses that distinction. Combinational circuits are built from logic, not prohibited from it; the feedback statement reverses the usual state relationship. Nor do combinational circuits inherently require a clock, while clocked sequential circuits update at clock events.

Why each choice is right or wrong

A - Incorrect: Combinational circuits are logic circuits; the statement incorrectly denies their ordinary implementation basis.

B - Correct: Present inputs determine combinational outputs, whereas sequential circuits additionally depend on retained state.

C - Incorrect: This reverses the state/feedback distinction; sequential storage can involve feedback, not its categorical absence.

D - Incorrect: Combinational logic does not inherently require a clock; clock events characterize clocked sequential updates.

Primary note proof

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Study-note section COA05

> A sequential circuit depends on stored state as well as inputs; registers, counters and flip-flops are examples.

Question sources

Computer Science Exit Exam · 2018 · Question 7

Question 133 | NS0304

Which one of the following is the first operation or step in CPU instruction cycle?

A. Executing the instruction

B. Handling interrupt

C. Decoding the instruction

D. Fetching the instruction

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Fetch the instruction.

COA15 begins the conceptual instruction cycle with fetch, then decode, operand acquisition, execution and any result write-back. Fetch must obtain the instruction before its opcode can be decoded and operation performed. Permitted interrupt checks are a separate stage/transition in that conceptual model, not the first ordinary instruction-processing step among these choices.

Why each choice is right or wrong

A - Incorrect: Execution follows obtaining and decoding the instruction, rather than being the cycle's first conceptual step.

B - Incorrect: Interrupt checks/service are distinct from the first ordinary fetch step in the note's conceptual cycle.

C - Incorrect: Decoding requires an already fetched instruction, so it follows rather than precedes fetch.

D - Correct: Fetching obtains the instruction and is explicitly first in the conceptual instruction-cycle sequence.

Primary note proof

Study-note section COA15

> A conceptual instruction cycle fetches an instruction, decodes opcode/addressing fields, obtains operands, executes, writes results where required and checks permitted interrupts.

Question sources

Computer Science Exit Exam · 2018 · Question 22

Question 134 | NS0310

_____ is a data transfer method in computer organization and architecture that allows an I/O device to transfer data directly to or from memory without the involvement of the CPU.

A. Asynchronous data transfer

B. Direct Memory Access (DMA)

C. Interrupt-driven transfer

D. Programmed I/O transfer

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Direct Memory Access (DMA).

COA30 says DMA lets a controller transfer a block directly between device and memory after CPU setup. The stem's without CPU involvement must mean without CPU moving every item, not no configuration/completion involvement whatsoever. Polling and interrupt-driven I/O still require CPU transfer work; asynchronous transfer describes timing coordination rather than this block-moving mechanism.

Why each choice is right or wrong

A - Incorrect: Asynchronous transfer describes coordination/timing rather than specifically controller-managed movement directly between device and memory.

B - Correct: DMA transfers the block through a controller after CPU setup, avoiding per-item CPU copying.

C - Incorrect: Interrupt-driven handlers still consume CPU time to service and move device data in this comparison.

D - Incorrect: Programmed I/O repeatedly checks status and has the CPU move each item rather than using DMA.

Primary note proof

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Question sources

Computer Science Exit Exam · 2018 · Question 31

Question 135 | NS0367

Which logic gate produces output 1 only when ALL inputs are 1?

A. OR gate

B. AND gate

C. NAND gate

D. XOR gate

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer B: AND gate.

COA03 states that AND is 1 only when all inputs are 1. This is exactly the condition in the stem. OR needs any high input, NAND complements AND, and two-input XOR is high when inputs differ. Thus none of the other truth functions has the specified all-high-only output rule.

Why each choice is right or wrong

A - Incorrect: OR becomes high when any input is high, so it does not require all inputs to be one.

B - Correct: AND outputs one exactly for the all-inputs-one combination required by the stem.

C - Incorrect: NAND inverts AND and therefore outputs zero, not one, when all its inputs are one.

D - Incorrect: Two-input XOR outputs one when inputs differ, not when both inputs are one.

Primary note proof

Study-note section COA03

> AND is 1 only when all inputs are 1; OR is 1 if any input is 1.

Question sources

Computer Organization & Architecture review questions · Question 1

Question 136 | NS0368

The Boolean expression F = A⊕B represents which gate?

A. XNOR gate

B. NOR gate

C. XOR gate

D. NAND gate

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: XOR gate.

The expression A⊕B is the exclusive-OR operation. COA03 defines its two-input output as 1 when the inputs differ and contrasts XNOR as agreement. NOR and NAND complement OR and AND instead. Thus XOR is the operation represented by the given symbol, with no extra circuit diagram or numeric assumption needed.

Why each choice is right or wrong

A - Incorrect: XNOR is the complement of XOR, giving one when the two inputs agree.

B - Incorrect: NOR complements OR; it is not the exclusive-OR operation denoted by the circled plus.

C - Correct: The circled-plus expression denotes exclusive OR, whose output is one for differing inputs.

D - Incorrect: NAND complements AND rather than implementing the given exclusive-OR expression.

Primary note proof

Study-note section COA-S06

> The circled plus symbol ⊕ denotes exclusive OR.

Study-note section COA03

> NAND complements AND; NOR complements OR.

Study-note section COA03

> For two inputs XOR is 1 when inputs differ, and XNOR when they agree.

Question sources

Computer Organization & Architecture review questions · Question 2

Question 137 | NS0369

Which gate is called the 'universal gate' because it can implement any Boolean function?

A. AND gate

B. OR gate

C. NAND gate

D. XOR gate

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: NAND, the only universal gate offered.

COA04 explicitly states that NAND and NOR are each universal and explains constructing NOT by tying inputs together, then building AND/OR combinations. Of those two universal families, only NAND is offered. AND and OR alone cannot provide inversion under this rule, and XOR is not identified as a universal family. Thus the correct offered gate is NAND.

Why each choice is right or wrong

A - Incorrect: AND alone is not the universal family described; it does not supply the required inversion construction.

B - Incorrect: OR alone is not the universal family described and cannot generate the stated NOT construction.

C - Correct: NAND is explicitly universal and can be combined to construct NOT, AND and OR.

D - Incorrect: XOR is not the NAND/NOR universal family; tied inputs produce zero rather than inversion.

Primary note proof

Study-note section COA04

> NAND and NOR are each universal: feeding the same input to both inputs constructs NOT; combine complemented gates to construct AND or OR.

Question sources

Computer Organization & Architecture review questions · Question 3

Question 138 | NS0370

For a NOR gate with inputs A=1, B=0, the output F is:

A. 0

B. 1

C. Undefined

D. Depends on voltage

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer A: F = 0.

COA03 defines OR as high when any input is high and NOR as its complement. At A=1, B=0, the OR value is 1, so NOR produces 0. The input combination is completely specified in ordinary Boolean logic; it is neither undefined nor dependent on an unspecified analog voltage model.

Why each choice is right or wrong

A - Correct: OR of 1 and 0 is 1; complementing that for NOR yields zero.

B - Incorrect: One is the OR result before inversion, not the NOR output for these inputs.

C - Incorrect: The Boolean NOR truth table fully defines this ordinary input combination.

D - Incorrect: The stem supplies Boolean input values, so no additional voltage threshold assumption is required.

Primary note proof

Study-note section COA03

> AND is 1 only when all inputs are 1; OR is 1 if any input is 1.

Study-note section COA03

> NAND complements AND; NOR complements OR.

Question sources

Computer Organization & Architecture review questions · Question 4

Question 139 | NS0371

De Morgan's theorem states that NOT(A·B) equals:

A. NOT(A) · NOT(B)

B. NOT(A) + NOT(B)

C. A + B

D. NOT(A+B)

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer B: NOT(A·B) = NOT(A) + NOT(B).

COA04 gives (AB)'=A'+B'. Complementing a product changes AND to OR and complements each operand. B matches that exact transformation. A instead gives the complement of A+B, C omits the operand complements, and D leaves the complement attached to a sum, which represents NOR rather than the requested NAND function.

Why each choice is right or wrong

A - Incorrect: The product of complements equals NOT(A+B), rather than the complement of the original product.

B - Correct: Complement each operand and change AND to OR, exactly as De Morgan’s product rule states.

C - Incorrect: A+B omits both required complements and is not generally equal to NOT(A·B).

D - Incorrect: NOT(A+B) is NOR and differs from NOT(A·B), for example when only one input is high.

Primary note proof

Study-note section COA04

> De Morgan gives (AB)'=A'+B' and (A+B)'=A'B'.

Question sources

Computer Organization & Architecture review questions · Question 5

Question 140 | NS0372

In Boolean algebra, the expression A + A' (A OR NOT A) always equals:

A. 0

B. A

C. 1

D. A'

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: A + A' = 1.

COA04 explicitly lists A+A'=1. If A is 0 then A' is 1, and if A is 1 then A' is 0; either way their Boolean OR is 1. Neither A nor A' remains fixed across both assignments, and zero is the product-of-complements result rather than their sum.

Why each choice is right or wrong

A - Incorrect: Zero is A AND NOT A; the given operation is OR, so one input is always high.

B - Incorrect: A can be zero while the expression remains one, so the identity is not simply A.

C - Correct: A bit OR its complement is always one for both possible assignments.

D - Incorrect: A complement can be zero while the full expression is one, so it is not the constant result.

Primary note proof

Study-note section COA04

> Useful identities are A+0=A, A·1=A, A+A=A, AA=A, A+A'=1, AA'=0, A+AB=A and A(A+B)=A.

Question sources

Computer Organization & Architecture review questions · Question 6

Question 141 | NS0373

How many rows are in the truth table for a Boolean function with 4 input variables?

A. 8

B. 12

C. 16

D. 32

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: 16 rows.

COA03 gives 2^n rows for n binary inputs. For four independent variables, each has two assignments, so the total combinations are 2×2×2×2=16. The options 8 and 32 correspond to three and five binary variables respectively; 12 does not enumerate all four-variable combinations.

Why each choice is right or wrong

A - Incorrect: Eight is 2^3, the number of assignments for three inputs rather than four.

B - Incorrect: Twelve omits four of the sixteen possible combinations of four independent binary inputs.

C - Correct: Four independent binary inputs yield 2^4=16 distinct truth-table rows.

D - Incorrect: Thirty-two is 2^5 and would require five independent binary input variables.

Primary note proof

Study-note section COA03

> With n binary inputs, enumerate 2^n truth-table rows.

Question sources

Computer Organization & Architecture review questions · Question 7

Question 142 | NS0374

For the Boolean function F=x+y'z with x=0,y=1,z=1, which options evaluate to the same Boolean value as F under that assignment? Select all that apply.

A. 0

B. 1

C. x

D. z

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer A, C: 0; x

Complement first: y is one, so y-prime is zero. The product y-prime AND z is therefore zero, and OR with x, also zero, leaves F equal to zero. Both the literal zero option and the expression x evaluate to that result under the stated assignment. The literal one and z both evaluate to one. The multiple-selection wording preserves the actual option values rather than arbitrarily rejecting an equivalent expression.

Why each choice is right or wrong

A - Correct: The literal zero equals the computed function value.

B - Incorrect: One differs from the computed zero.

C - Correct: x is assigned zero and therefore equals the function value.

D - Incorrect: z is assigned one and differs from the function value.

Primary note proof

Study-note section COA03

> Evaluate complements, products and then sums.

Study-note section COA03

> A plus sign means Boolean OR, not integer addition.

Study-note section COA04

> Useful identities are A+0=A, A·1=A, A+A=A, AA=A, A+A'=1, AA'=0, A+AB=A and A(A+B)=A.

Question sources

Computer Organization & Architecture review questions · Question 8

Question 143 | NS0375

Which characteristic best describes a flip-flop?

A. A combinational logic circuit

B. A bistable multivibrator that stores one bit

C. A 4-bit counter

D. An arithmetic unit

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer B: A bistable one-bit storage element.

COA05 says a flip-flop stores one bit while powered and classifies flip-flops as sequential circuits with stored state. B is the option describing that storage element. A combinational circuit lacks stored logical state; a four-bit counter requires a multi-bit state structure, and an arithmetic unit computes rather than defining this one-bit storage component.

Why each choice is right or wrong

A - Incorrect: A combinational circuit has no stored logical state, whereas a flip-flop retains a bit.

B - Correct: Bistable one-bit storage matches the powered state-retention role of a flip-flop.

C - Incorrect: One flip-flop stores one bit; a four-bit counter requires a multi-bit state organization.

D - Incorrect: An arithmetic unit computes values rather than defining the one-bit sequential storage element.

Primary note proof

Study-note section COA-S07

> A flip-flop is bistable: it has two stable states representing 0 and 1.

Study-note section COA-S07

> Multiple flip-flops are required to represent a multi-bit counter state.

Study-note section COA05

> A combinational output depends on present inputs after propagation delay.

Question sources

Computer Organization & Architecture review questions · Question 9

Question 144 | NS0376

What distinguishes a flip-flop from a simple logic gate in terms of memory?

A. Flip-flops use more power

B. Flip-flops retain state even after inputs change

C. Flip-flops operate faster

D. Flip-flops require AC power

Hint

Check whether output depends only on current inputs or also on stored state, and track the stated clock behavior.

Answer B: Flip-flops retain stored state during powered operation.

COA05 distinguishes sequential state from combinational outputs determined by present inputs. A flip-flop can retain its bit when inputs change without a qualifying update, giving the memory distinction in B. This does not imply nonvolatile retention after power-off. Power, speed and AC-supply claims are neither the defining property nor guarantees established here.

Why each choice is right or wrong

A - Incorrect: Power consumption is implementation-dependent and does not define whether a circuit has logical memory.

B - Correct: Stored state persists between valid updates while powered, unlike a purely present-input combinational gate.

C - Incorrect: Speed is not the defining memory distinction and is not guaranteed by the flip-flop label.

D - Incorrect: The note specifies powered storage, not a requirement for an AC rather than another supply.

Primary note proof

Study-note section COA05

> A flip-flop stores one bit while powered.

Study-note section COA05

> State retention during operation does not mean nonvolatile storage.

Question sources

Computer Organization & Architecture review questions · Question 10

Question 145 | NS0377

What is the decimal equivalent of binary number 1011?

A. 9

B. 10

C. 11

D. 13

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer C: $1011_2 = 11_{10}$.

Using COA08's base-position rule, 1011_2 has weights 8, 4, 2 and 1. Its high bits contribute 8+2+1=11, while the 4-place is zero. The note's worked example confirms the same sum. Nine, ten and thirteen correspond to different low-order bit patterns, not the supplied 1011.

Why each choice is right or wrong

A - Incorrect: Nine equals 8+1, corresponding to 1001_2 and omitting the given two-place bit.

B - Incorrect: Ten equals 8+2, corresponding to 1010_2 and omitting the given one-place bit.

C - Correct: The set bit positions contribute 8+2+1=11 in decimal.

D - Incorrect: Thirteen equals 8+4+1, corresponding to 1101_2 rather than the supplied 1011_2.

Primary note proof

Study-note section COA08

> In base b, a digit at position i has value digit×b^i.

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Question sources

Computer Organization & Architecture review questions · Question 11

Question 146 | NS0378

In hexadecimal, what does the letter 'F' represent in decimal?

A. 14

B. 15

C. 16

D. 10

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer B: $F_{16} = 15_{10}$.

COA08 defines hexadecimal digits as 0-9,A-F and explicitly gives F=15. These sixteen symbols represent values zero through fifteen, so sixteen itself requires the two-digit hexadecimal representation 10. Fourteen is E, and ten is A, leaving fifteen as the only value of the single digit F.

Why each choice is right or wrong

A - Incorrect: Fourteen corresponds to the preceding hexadecimal digit E, not the final digit F.

B - Correct: The note explicitly assigns decimal value fifteen to the hexadecimal digit F.

C - Incorrect: Sixteen requires hexadecimal 10; it is outside the value range of a single hexadecimal digit.

D - Incorrect: Ten corresponds to hexadecimal A rather than the later digit F.

Primary note proof

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Question sources

Computer Organization & Architecture review questions · Question 12

Question 147 | NS0379

The octal number system uses a base of:

A. 2

B. 8

C. 10

D. 16

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer B: Base 8.

COA08 gives octal digits 0-7: eight possible digit values, so octal has base eight. Binary has two digits, decimal ten and hexadecimal sixteen. Place weights in octal are therefore powers of eight, not powers of two, ten or sixteen, even though octal digits can be grouped with three binary bits.

Why each choice is right or wrong

A - Incorrect: Base two is binary; octal allows eight digit values rather than only zero and one.

B - Correct: Octal uses digits zero through seven, giving a base of eight.

C - Incorrect: Base ten is decimal, not the eight-digit octal number system.

D - Incorrect: Base sixteen is hexadecimal, which uses the additional A-F digits absent from octal.

Primary note proof

Study-note section COA08

> In base b, a digit at position i has value digit×b^i.

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Question sources

Computer Organization & Architecture review questions · Question 13

Question 148 | NS0380

Which number system is most fundamental to digital computing and why?

A. Decimal, because humans use it

B. Octal, because it maps to 3 bits

C. Binary, because digital circuits have two states (on/off)

D. Hexadecimal, because it is compact

Hint

Apply the gate truth rules and Boolean identities in the stated input order; distinguish an inversion from the underlying operation.

Answer C: Binary matches the two logical states used by digital logic.

COA03 defines digital gates using the two Boolean values 0 and 1, and COA08 defines binary using exactly those two digits. That direct correspondence explains why binary is fundamental to the digital representation described here. Octal and hexadecimal group binary bits for convenience, while decimal's human familiarity does not supply the two-state circuit correspondence.

Why each choice is right or wrong

A - Incorrect: Human decimal familiarity does not match the two Boolean states used here's digital gates.

B - Incorrect: Octal groups three binary bits and is a convenient notation rather than the underlying two-state representation.

C - Correct: Binary digits zero and one directly correspond to the two logical values used by digital circuits.

D - Incorrect: Hexadecimal compactly groups four binary bits; compact notation does not replace the underlying binary states.

Primary note proof

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Study-note section COA03

> NOT complements a bit.

Question sources

Computer Organization & Architecture review questions · Question 14

Question 149 | NS0381

How many bits are needed to represent the hexadecimal digit F?

A. 2

B. 3

C. 4

D. 8

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer C: 4 bits: $F_{16} = 1111_2$.

COA08 groups four binary bits for each hexadecimal digit and gives F=15. COA09's unsigned range shows three bits reach only 7, while four bits reach 15. Therefore the minimum representation of F is four bits, 1111. Eight bits can contain the value but are not the number needed for one hexadecimal digit.

Why each choice is right or wrong

A - Incorrect: Two bits encode only zero through three, insufficient for hexadecimal F's decimal value fifteen.

B - Incorrect: Three bits encode only zero through seven, so they cannot represent the value fifteen.

C - Correct: Four bits encode zero through fifteen; F is represented by the four-bit pattern 1111.

D - Incorrect: Eight bits could contain fifteen but exceed the minimum four bits needed for a hexadecimal digit.

Primary note proof

Study-note section COA08

> Group binary bits in threes for octal and fours for hexadecimal, outward from the radix point.

Study-note section COA08

> Binary uses 0-1, octal 0-7 and hexadecimal 0-9,A-F; F=15.

Study-note section COA09

> For n-bit unsigned values the range is 0 through 2^n−1.

Question sources

Computer Organization & Architecture review questions · Question 15

Question 150 | NS0382

Which listed memory categories lose their contents when power is removed? Assume ordinary SRAM/DRAM RAM and an SRAM-based processor cache. Select all that apply.

A. ROM

B. Flash memory

C. RAM

D. Cache

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C, D: RAM; Cache

Ordinary SRAM and DRAM require power to retain their contents. RAM therefore qualifies, and the stated SRAM-based cache qualifies for the same technological reason. Cache is a hierarchy role rather than a promise of nonvolatile storage. ROM and flash retain contents without power, so they are excluded. Specifying implementation makes the volatility judgment definite while keeping both valid categories rather than forcing a RAM-only answer.

Why each choice is right or wrong

A - Incorrect: ROM retains contents without power.

B - Incorrect: Flash retains contents without power.

C - Correct: Ordinary SRAM/DRAM RAM is volatile.

D - Correct: The explicitly SRAM-based cache is volatile too.

Primary note proof

Study-note section COA24

> SRAM uses powered bistable storage: fast and costly per bit, without periodic refresh.

Study-note section COA24

> DRAM stores charge and requires refresh: denser and commonly used for main memory.

Study-note section COA24

> Both are ordinarily volatile.

Study-note section COA24

> ROM and flash retain contents without power; erasable/programmed variants mean “ROM can never be modified” is too broad.

Question sources

Computer Organization & Architecture review questions · Question 16

Question 151 | NS0383

Cache memory is best described as:

A. Large, slow, cheap memory

B. Small, ultra-fast memory between CPU and RAM

C. Non-volatile backup storage

D. Virtual memory on disk

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Small, fast storage near the CPU between it and main memory.

COA23 places cache above DRAM in the usual speed/cost hierarchy and says faster levels are generally smaller and more expensive per bit. It defines cache as copied memory blocks near the CPU. B captures that role. Cache is not the large slow backup layer or disk-based virtual-memory backing store; its useful speed derives from proximity/reuse, not guaranteed persistence.

Why each choice is right or wrong

A - Incorrect: Large slow cheap storage describes lower hierarchy levels rather than the small fast cache layer.

B - Correct: Cache is a small fast near-CPU copy of memory blocks used to accelerate access.

C - Incorrect: The note defines cache for accelerated reuse, not as nonvolatile backup storage.

D - Incorrect: Disk-backed virtual memory is a different address-space/backing-store mechanism from the CPU cache.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA23

> Faster levels are generally smaller and more expensive per bit.

Study-note section COA23

> Cache copies memory blocks near the CPU.

Question sources

Computer Organization & Architecture review questions · Question 17

Question 152 | NS0384

In direct memory mapping, what happens when two different main memory blocks map to the same cache slot?

A. Both are stored simultaneously

B. A cache conflict occurs and one block replaces the other

C. The system crashes

D. The blocks merge

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: A placement conflict: only one block can occupy that direct-mapped slot.

COA25 states that two blocks with the same direct-mapped index cannot reside together. When the requested block differs from the valid block at that slot, the cache must replace the slot's contents, producing a conflict miss. This is ordinary cache operation, not a crash or merging of block contents; direct mapping offers no alternate placement for the same index.

Why each choice is right or wrong

A - Incorrect: One direct-mapped line holds one block at a time, so the colliding blocks cannot coexist there.

B - Correct: The shared index causes a placement conflict; installing the requested block replaces the current one.

C - Incorrect: Conflict replacement is ordinary cache behavior and does not imply a system crash.

D - Incorrect: Cache replacement preserves separate memory-block identities; it does not merge their contents into one block.

Primary note proof

Study-note section COA25

> Direct mapping uses line=block mod L.

Study-note section COA25

> Two blocks with the same direct-mapped index cannot reside together.

Question sources

Computer Organization & Architecture review questions · Question 18

Question 153 | NS0385

Which cache mapping technique offers the most flexibility in placing data blocks?

A. Direct mapping

B. Set associative mapping

C. Fully associative mapping

D. Sequential mapping

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Fully associative mapping

COA25 says direct mapping fixes one line, set associativity permits any way within one indexed set, and fully associative placement permits any line. Allowing every line gives the most placement flexibility for a fixed cache. The sequential-mapping option is not one of the cache placement rules defined by the supplied note.

Why each choice is right or wrong

A - Incorrect: Direct mapping restricts a block to a single line and therefore has the least placement freedom.

B - Incorrect: Set associativity permits several ways in one set, fewer locations than the entire fully associative cache.

C - Correct: Fully associative placement allows any cache line, maximizing placement freedom for each block.

D - Incorrect: Sequential ordering is not a placement freedom comparable to the defined direct, set-associative and fully associative cache rules.

Primary note proof

Study-note section COA25

> Direct mapping uses line=block mod L.

Study-note section COA25

> Fully associative placement permits any line.

Study-note section COA25

> A W-way set-associative cache has S=L/W sets and set=block mod S; any of W ways in that set may hold the block.

Question sources

Computer Organization & Architecture review questions · Question 19

Question 154 | NS0386

Set associative mapping is described as a compromise between:

A. RAM and ROM

B. Direct and fully associative mapping

C. Primary and secondary memory

D. Cache and virtual memory

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: Direct and fully associative mapping.

COA25 defines W-way set associativity as fixing a set by the block index while allowing any of W ways inside it. That combines direct indexing with an associative choice within the selected set, placing it between one-line direct mapping and unrestricted fully associative placement. RAM/ROM and primary/secondary are technology/hierarchy contrasts, not the two placement endpoints.

Why each choice is right or wrong

A - Incorrect: RAM and ROM describe memory types rather than the placement-policy endpoints combined by set associativity.

B - Correct: Set indexing restricts placement like direct mapping while allowing an associative choice among the selected set's ways.

C - Incorrect: Primary and secondary describe hierarchy roles, not the direct versus associative cache-placement alternatives.

D - Incorrect: Cache and virtual memory are different mechanisms rather than the two placement extremes defining set associativity.

Primary note proof

Study-note section COA25

> Direct mapping uses line=block mod L.

Study-note section COA25

> Fully associative placement permits any line.

Study-note section COA25

> A W-way set-associative cache has S=L/W sets and set=block mod S; any of W ways in that set may hold the block.

Question sources

Computer Organization & Architecture review questions · Question 20

Question 155 | NS0387

Virtual memory primarily uses which storage device to extend the apparent memory size?

A. ROM

B. Cache

C. Hard disk / SSD

D. Registers

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Hard disk / SSD as backing storage.

COA27 says backing storage can hold nonresident pages while virtual memory provides translated protected address spaces. COA28 identifies HDDs and SSDs as persistent storage devices, matching that backing role. ROM, cache and registers are not the nonresident-page backing tier described. Apparent extension does not mean physical RAM is added or that virtual memory is only disk disguised as RAM.

Why each choice is right or wrong

A - Incorrect: ROM stores persistent firmware/data but is not the writable nonresident-page backing role described here.

B - Incorrect: Cache holds near-CPU copies of memory blocks rather than serving as the mass backing store.

C - Correct: Persistent HDD/SSD storage can hold nonresident page contents for the virtual-memory system.

D - Incorrect: Registers hold immediate execution state and cannot serve as the large backing store for nonresident pages.

Primary note proof

Study-note section COA27

> Virtual memory gives processes translated address spaces and protection; backing storage can hold nonresident pages.

Study-note section COA28

> An HDD stores magnetic patterns on tracks divided into sectors; reads/writes access addressed blocks while platters rotate.

Study-note section COA28

> SSDs use nonvolatile semiconductor storage without mechanical seek.

Question sources

Computer Organization & Architecture review questions · Question 21

Question 156 | NS0388

A valid disk-backed virtual page is currently nonresident, and the access must load its saved contents before continuing. Which condition describes the resulting page-fault case?

A. The CPU runs out of registers

B. The requested page is not in physical memory and must be loaded from disk

C. Cache memory is full

D. An arithmetic overflow occurs

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: The requested page is not in physical memory and must be loaded from disk

Accessing the valid but nonresident disk-backed page faults so that the system can make the page resident and continue. The stated case requires loading its saved contents. Register pressure, cache replacement and arithmetic overflow are different conditions. Other kinds of page fault need not read a disk, so the question explicitly limits itself to this nonresident disk-backed case.

Why each choice is right or wrong

A - Incorrect: Register allocation pressure is not an access to a missing or forbidden virtual page.

B - Correct: The described disk-backed nonresident access is a page-fault case, although not all faults require a disk read.

C - Incorrect: A full cache leads to cache replacement, not necessarily a virtual-memory page fault.

D - Incorrect: Arithmetic overflow is an instruction arithmetic condition rather than a virtual-page access event.

Primary note proof

Study-note section COA27

> A valid nonresident page access faults and may require loading data; invalid or forbidden accesses can also fault.

Study-note section COA27

> A cache miss, TLB miss and page fault are different events; not every page fault reads a disk, since zero-fill and already-resident shared pages exist.

Question sources

Computer Organization & Architecture review questions · Question 22

Question 157 | NS0389

Compared to RAM, a Hard Disk Drive (HDD) is:

A. Faster and volatile

B. Slower and non-volatile

C. Faster and non-volatile

D. Slower and volatile

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer B: HDD is generally slower and nonvolatile.

COA23 places DRAM above HDD in the usual speed hierarchy. COA28 says an HDD does not require continuous power to preserve stored bits, while COA24 calls ordinary SRAM/DRAM volatile. Combining the independent speed and retention rules gives slower and nonvolatile. The answer concerns the usual hierarchy comparison rather than every conceivable workload or specialized memory device.

Why each choice is right or wrong

A - Incorrect: It reverses both usual properties: HDD is slower than RAM and retains data without power.

B - Correct: The hierarchy places HDD below RAM in speed, and magnetic storage preserves bits without continuous power.

C - Incorrect: Nonvolatility is right, but the claim that HDD is faster than RAM reverses the usual hierarchy.

D - Incorrect: Slower is right, but volatile contradicts the note's explicit HDD power-off retention statement.

Primary note proof

Study-note section COA23

> A usual speed/cost-per-bit order is registers, L1/L2/L3 cache, DRAM, SSD, HDD and archival tape.

Study-note section COA24

> Both are ordinarily volatile.

Study-note section COA28

> It does not need continuous power to preserve stored bits.

Question sources

Computer Organization & Architecture review questions · Question 23

Question 158 | NS0390

Which range matches the historical optical-disk access-time examples used in this course, rather than a universal specification for every optical drive?

A. 1-5 milliseconds

B. 10-30 milliseconds

C. 100-300 milliseconds

D. 500-1000 milliseconds

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: 100-300 milliseconds

The note identifies one hundred to three hundred milliseconds as a historical optical-access example. That range is not a universal specification: actual timing depends on the device and measurement conditions. The revised stem asks for the particular teaching range rather than an unqualified typical value for all optical disks. The remaining intervals do not match that historical example and therefore are definite distractors.

Why each choice is right or wrong

A - Incorrect: This interval does not match the historical optical-access example.

B - Incorrect: This shorter interval is not the course's cited historical example.

C - Correct: This is the historical example range given in the note.

D - Incorrect: This longer interval is not the course's cited historical example.

Primary note proof

Study-note section COA28

> Optical access ranges such as 100-300 ms are historical examples, not universal device specifications.

Question sources

Computer Organization & Architecture review questions · Question 24

Question 159 | NS0391

Which storage medium accesses data sequentially rather than randomly?

A. SSD

B. HDD

C. Magnetic tape

D. Optical disk

Hint

Separate storage role, access timing, placement and power retention; use the specified hierarchy or mapping convention.

Answer C: Magnetic tape.

COA28 classifies tape as sequential and suited to streaming/archives. Disk provides direct addressed-block access, while SSD provides nonmechanical persistent storage rather than a tape-style stream. The question asks the medium whose access is characteristically sequential, so magnetic tape is the supported choice. Physical streaming inside another device does not make its logical access interface tape-like.

Why each choice is right or wrong

A - Incorrect: SSD is addressed nonvolatile semiconductor storage rather than the sequential tape-access medium described.

B - Incorrect: HDD permits addressed-block direct access rather than requiring tape-like sequential record traversal.

C - Correct: Tape is explicitly the sequential streaming/archive medium in the note's access taxonomy.

D - Incorrect: Optical disk uses addressed disk access rather than being the sequential tape medium requested.

Primary note proof

Study-note section COA28

> Optical media use lasers; tape is suited to sequential streaming and archives.

Study-note section COA28

> In textbook access taxonomy, tape is sequential, disk is direct access with positioning delay, RAM is random access with broadly address-independent timing, and associative access matches content.

Question sources

Computer Organization & Architecture review questions · Question 25

Question 160 | NS0392

Registers within the CPU are used primarily to:

A. Store large datasets permanently

B. Hold temporary data for immediate CPU operations

C. Communicate with I/O devices

D. Manage virtual memory pages

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Hold temporary values for immediate CPU operations.

COA14 distinguishes registers as explicit execution operands and assigns them fetched instructions, transferred data and operands/results. That supports the immediate temporary-storage role in B. Registers are small execution state rather than permanent bulk data storage. Some registers participate in I/O or translation, but those specialized uses do not replace the primary execution-storage role asked here.

Why each choice is right or wrong

A - Incorrect: Registers are small temporary execution storage rather than permanent bulk storage for large datasets.

B - Correct: Registers hold operands, instruction state and intermediate results used directly by current CPU operations.

C - Incorrect: Some registers support I/O, but general I/O communication is not their primary execution-storage purpose.

D - Incorrect: Translation can use specialized state, but managing virtual pages is not the primary role of CPU registers generally.

Primary note proof

Study-note section COA14

> IR holds the fetched instruction; MAR holds an address used for the current memory access; MDR holds transferred data.

Study-note section COA14

> The accumulator holds operands/results, SP tracks the stack, and status flags report conditions.

Study-note section COA14

> Registers and cache both store temporary information, but registers are explicit execution operands and cache holds copies of memory blocks.

Question sources

Computer Organization & Architecture review questions · Question 26

Question 161 | NS0393

The address bus in a computer system is responsible for:

A. Carrying actual data between components

B. Specifying the memory location the CPU wants to access

C. Providing timing signals

D. Transmitting control commands

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Specify the addressed memory location.

COA12 says address signals select a location, data signals carry values and control signals coordinate read/write and timing. Therefore B directly matches the address bus. Actual payload data belongs to the data bus, while timing and operation commands belong to control signaling. The distinction does not depend on address width or the value being transferred.

Why each choice is right or wrong

A - Incorrect: Actual payload values travel on the data bus rather than the address signals selecting a location.

B - Correct: The address bus identifies the memory location selected for the CPU's access.

C - Incorrect: Timing belongs to control signaling, not the location-selection role of address signals.

D - Incorrect: Read/write and other operation commands belong to control signaling rather than the address bus.

Primary note proof

Study-note section COA12

> Address signals select a location; data signals carry values; control signals coordinate read/write, timing, requests and acknowledgements.

Question sources

Computer Organization & Architecture review questions · Question 27

Question 162 | NS0394

A CPU exposes 32 address bits and uses byte-addressable memory with no additional address-extension mechanism. What maximum byte capacity can it address? Use GB here to mean 2^30 bytes.

A. 2 GB

B. 4 GB

C. 8 GB

D. 16 GB

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: 4 GB

Thirty-two address bits identify 2^32 distinct locations. With one byte per location, that is 2^32 bytes, or four units of 2^30 bytes. The selected option is therefore four GB under the explicitly stated binary convention, more precisely four GiB. Data-bus width is irrelevant to this location count, and without byte addressing the same number of address bits would not determine a byte capacity.

Why each choice is right or wrong

A - Incorrect: Two binary gigabytes would require only 2^31 byte locations.

B - Correct: 2^32 byte locations equal four units of 2^30 bytes.

C - Incorrect: Eight binary gigabytes would require 33 address bits.

D - Incorrect: Sixteen binary gigabytes would require 34 address bits.

Primary note proof

Study-note section COA12

> With a address bits there are 2^a addressable locations.

Study-note section COA12

> Multiply by bytes per location before calling this a byte capacity.

Question sources

Computer Organization & Architecture review questions · Question 28

Question 163 | NS0395

The control bus in a computer system carries:

A. Data between CPU and memory

B. Memory addresses

C. Signals that regulate operations like read/write timing

D. Interrupt requests only

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: Control signals coordinate operations, including read/write timing.

COA12 defines control signals as coordinating read/write, timing, requests and acknowledgements. C expresses that general role. The data and address alternatives belong to distinct signal groups. Interrupt requests may be part of aggregate control, but the word only makes D too restrictive because read/write and timing controls are also included.

Why each choice is right or wrong

A - Incorrect: Actual transferred data belongs to the data bus rather than the control-signal group.

B - Incorrect: Location selection belongs to address signals rather than the control bus's coordination role.

C - Correct: Read/write coordination and timing are explicitly among the functions assigned to control signals.

D - Incorrect: Interrupt requests are only one possible control input; the bus also carries other coordination signals.

Primary note proof

Study-note section COA12

> Address signals select a location; data signals carry values; control signals coordinate read/write, timing, requests and acknowledgements.

Study-note section COA12

> Aggregate control signals include both CPU outputs and device inputs; both data and control buses can therefore be bidirectional under that scope.

Question sources

Computer Organization & Architecture review questions · Question 29

Question 164 | NS0396

In instruction format terminology, an 'opcode' refers to:

A. The memory address of the next instruction

B. The bits specifying the operation to perform

C. The register holding the result

D. The data value being processed

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Bits specifying the operation.

COA16 defines the opcode as the operation-specifying field. Operand fields locate registers/data and mode fields explain their interpretation. Thus B names the correct format role; next-instruction addresses, result-register identity and operand values are separate quantities. Their presence elsewhere in a machine does not change what opcode means.

Why each choice is right or wrong

A - Incorrect: Next-instruction position is a PC/control-flow role, not the opcode's operation-selection role.

B - Correct: Opcode bits select the operation the processor should perform for the instruction.

C - Incorrect: Result-register selection belongs to an operand/destination field rather than defining the opcode.

D - Incorrect: An operand data value is distinct from the code selecting the operation to apply.

Primary note proof

Study-note section COA16

> An opcode specifies the operation; operand fields locate data or registers, and a mode field explains interpretation.

Question sources

Computer Organization & Architecture review questions · Question 30

Question 165 | NS0397

A zero-address instruction format relies on:

A. Implicit register addressing

B. Stack-based operand access

C. Immediate values only

D. Direct memory addressing

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: Implicit top-of-stack operands.

COA16 says zero-address stack ADD uses two implicit top-of-stack operands. The absence of written operand addresses therefore relies on stack organization, not immediate constants or direct memory fields. Generic implicit register addressing is too broad and does not identify the stack discipline specified for this format; PUSH/POP may still have explicit addresses.

Why each choice is right or wrong

A - Incorrect: Implicit register addressing alone does not identify the stack operand convention defining zero-address arithmetic here.

B - Correct: Operands come implicitly from the top of the stack, so arithmetic needs no explicit operand-address fields.

C - Incorrect: Immediate values require encoded operand information; zero-address stack arithmetic is not restricted to constants.

D - Incorrect: Direct memory addressing would use an explicit operand address, contrary to the stated zero-address arithmetic format.

Primary note proof

Study-note section COA16

> Zero-address stack ADD uses two implicit top-of-stack operands.

Study-note section COA17

> PUSH and POP may still contain an explicit address even on a stack machine.

Question sources

Computer Organization & Architecture review questions · Question 31

Question 166 | NS0398

Which instruction format is most expressive, allowing operations on three separate operands?

A. Zero-address

B. One-address

C. Two-address

D. Three-address

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer D: Three-address format.

COA16 defines three-address ADD X,Y,Z as X←Y+Z, allowing a separately named destination and two sources. This matches the stem's three separate operands. Two-address form overwrites one source, one-address uses an implicit accumulator, and zero-address uses stack operands. Most expressive is interpreted here through the explicit three-operand criterion rather than a universal comparison of complete instruction sets.

Why each choice is right or wrong

A - Incorrect: Zero-address arithmetic names no explicit operands and instead uses the implicit stack top.

B - Incorrect: One-address accumulator arithmetic names only one explicit operand and relies on an implicit accumulator.

C - Incorrect: Two-address arithmetic overwrites a named source and lacks a separately named third operand field.

D - Correct: Three explicit fields allow one destination and two independently named source operands.

Primary note proof

Study-note section COA16

> Three-address ADD X,Y,Z means X←Y+Z under destination-first notation.

Study-note section COA16

> Two-address ADD X,Y overwrites X with X+Y.

Study-note section COA16

> One-address accumulator ADD X means AC←AC+M[X].

Study-note section COA16

> Zero-address stack ADD uses two implicit top-of-stack operands.

Question sources

Computer Organization & Architecture review questions · Question 32

Question 167 | NS0399

Stack organization follows which principle?

A. FIFO (First In First Out)

B. LIFO (Last In First Out)

C. Random Access

D. Priority-based Access

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer B: LIFO: last in, first out.

COA17 defines a stack as last-in, first-out: PUSH adds at the logical top and POP removes that top item. The most recently pushed remaining item is therefore the next removed. FIFO describes queue ordering, while arbitrary indexed access or priority selection does not define the stack discipline.

Why each choice is right or wrong

A - Incorrect: FIFO removes the oldest inserted item, the opposite order from normal stack removal.

B - Correct: The last item pushed becomes the top and is the first item popped.

C - Incorrect: Random access does not enforce removal of the most recently pushed remaining item.

D - Incorrect: Stack removal follows insertion order in reverse, not a priority ranking assigned to items.

Primary note proof

Study-note section COA17

> A stack is last-in, first-out.

Study-note section COA17

> PUSH adds an item and POP removes the logical top.

Question sources

Computer Organization & Architecture review questions · Question 33

Question 168 | NS0400

In indirect addressing mode, the instruction contains:

A. The actual data value

B. The direct memory address of the data

C. A pointer to the memory address where data resides

D. A register number

Hint

Identify the register, signal or instruction role and the execution phase; distinguish stored values, addresses and control actions.

Answer C: A reference whose stored contents supply the effective address.

COA18 defines memory-indirect addressing as EA=M[A], followed by operand M[EA]. Thus the instruction's field identifies a pointer location rather than directly containing the operand value or final operand address. C is the pointer/indirection description. A describes immediate, B direct, and a plain register number identifies a register-based mode rather than this memory-indirect convention.

Why each choice is right or wrong

A - Incorrect: An actual value encoded in the instruction is immediate addressing rather than memory indirection.

B - Incorrect: Using the field directly as the operand's address is direct addressing, without the intermediate pointer lookup.

C - Correct: The field identifies a pointer-bearing location; its contents supply the address of the operand data.

D - Incorrect: A register number names register-based addressing rather than the memory-indirect field defined here.

Primary note proof

Study-note section COA18

> Let A be an instruction field, R a register value and M[x] the contents at x.

Study-note section COA18

> Immediate uses value A; direct uses EA=A and operand M[A]; indirect uses EA=M[A] then operand M[EA].

Study-note section COA18

> Register uses operand R; register-indirect uses EA=R.

Question sources

Computer Organization & Architecture review questions · Question 34

Question 169 | NS0401

Which is a textbook RISC design tendency, without claiming every instruction has single-cycle latency?

A. Variable-length instructions

B. Large instruction set (100-250 ops)

C. Simple operations, regular encodings and load/store memory access

D. Complex microcode control unit

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer C: Simple operations, regular encodings and load/store memory access

Textbook RISC emphasizes simple operations, regular encodings, load/store memory access and pipeline-friendly control. These tendencies do not imply every instruction completes with one-cycle latency; steady-state pipeline throughput is a separate measure. Variable lengths and complex operations are associated with CISC tendencies, while an instruction-count range is not a defining guarantee. The choice states these RISC characteristics without imposing a universal single-cycle latency.

Why each choice is right or wrong

A - Incorrect: Variable lengths are associated with CISC tendencies rather than the stated RISC summary.

B - Incorrect: This numerical instruction-count range is not the supported RISC characteristic.

C - Correct: These are the RISC design tendencies named in the note.

D - Incorrect: Complex microcode control is not the stated RISC design tendency.

Primary note proof

Study-note section COA20

> Textbook RISC emphasizes simple operations, regular encodings, load/store memory access, plentiful registers and pipeline-friendly control.

Study-note section COA20

> CISC commonly supplies more complex operations, richer addressing modes and variable lengths.

Study-note section COA20

> One instruction completing per pipeline cycle is throughput, not proof that every instruction has one-cycle latency.

Question sources

Computer Organization & Architecture review questions · Question 35

Question 170 | NS0402

CISC processors differ from RISC processors primarily in that CISC:

A. Has fewer instructions

B. Uses only register-to-register operations

C. Supports complex instructions that may take multiple cycles

D. Has no microcode

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer C: CISC commonly supports richer complex instructions, potentially with multi-cycle execution.

COA20 associates CISC with complex operations, richer addressing and variable lengths, and rejects universal one-cycle latency claims. C therefore gives the matching tendency and permits, rather than requires, multi-cycle execution. The remaining claims either reverse the usual complexity contrast or impose unsupported only/no restrictions. The answer does not claim modern RISC instructions can never take multiple cycles.

Why each choice is right or wrong

A - Incorrect: Fewer instructions reverses the usual simple-versus-rich instruction-set contrast rather than stating the note's CISC tendency.

B - Incorrect: Only register-to-register operations contradicts the richer addressing modes associated with CISC in the note.

C - Correct: Complex operations and possible multi-cycle latency fit the CISC tendency without claiming exclusivity of multi-cycle execution.

D - Incorrect: The note does not define CISC by absence of microcode; control implementation is a separate design dimension.

Primary note proof

Study-note section COA20

> Textbook RISC emphasizes simple operations, regular encodings, load/store memory access, plentiful registers and pipeline-friendly control.

Study-note section COA20

> CISC commonly supplies more complex operations, richer addressing modes and variable lengths.

Study-note section COA20

> Modern RISC instructions can take multiple cycles and modern CISC processors can translate instructions into simpler internal operations.

Question sources

Computer Organization & Architecture review questions · Question 36

Question 171 | NS0403

A RISC processor typically has more _____ compared to a CISC processor to compensate for simpler instructions:

A. Clock cycles per instruction

B. Addressing modes

C. Registers

D. Instruction bytes

Hint

Compare textbook design tendencies without turning them into guarantees about every processor.

Answer C: Registers.

COA20 explicitly lists plentiful registers among textbook RISC characteristics, alongside load/store access and simple operations. C is therefore the supported tendency. More addressing modes and instruction bytes are not the stated compensating mechanism, and more cycles per instruction is not a defining guarantee. The typically qualifier matters because the labels alone do not impose a register count on every implementation.

Why each choice is right or wrong

A - Incorrect: More cycles per instruction is not the plentiful-register mechanism or a guaranteed RISC property.

B - Incorrect: Richer addressing modes are associated with CISC tendencies, not the stated RISC compensating feature.

C - Correct: Plentiful registers are explicitly one of the textbook RISC traits named by the complete note.

D - Incorrect: More instruction bytes is not the feature identified as supporting simple RISC operations.

Primary note proof

Study-note section COA20

> Textbook RISC emphasizes simple operations, regular encodings, load/store memory access, plentiful registers and pipeline-friendly control.

Study-note section COA-S04

> RISC means Reduced Instruction Set Computer; CISC means Complex Instruction Set Computer.

Question sources

Computer Organization & Architecture review questions · Question 37

Question 172 | NS0404

In Programmed I/O, how does the CPU know when the I/O device has finished its operation?

A. The I/O device sends an interrupt signal

B. The CPU periodically polls (checks) the status register

C. DMA controller notifies the CPU

D. The operating system schedules a check

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Poll the status register.

COA30 defines programmed/polled I/O as repeatedly checking device status while the CPU moves each item. B names that readiness/completion checking mechanism. Interrupt notification belongs to interrupt-driven I/O, and a DMA controller is a separate transfer mechanism. The OS may arrange when polling code runs, but that broad scheduling statement does not replace the actual status check.

Why each choice is right or wrong

A - Incorrect: An interrupt signal names interrupt-driven notification rather than programmed I/O's polling mechanism.

B - Correct: The CPU reads device status repeatedly to determine readiness or completion in programmed I/O.

C - Incorrect: DMA-controller completion notification belongs to a DMA transfer rather than programmed CPU-driven I/O.

D - Incorrect: Scheduling a check is broader software organization; polling the status register is the actual completion-detection method.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Question sources

Computer Organization & Architecture review questions · Question 38

Question 173 | NS0405

The main advantage of Interrupt-Driven I/O over Programmed I/O is:

A. Higher data transfer speeds

B. The CPU can perform other tasks while waiting for I/O

C. It requires no special hardware

D. It eliminates the need for device drivers

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: The CPU can do other work while the device operates.

COA30 says interrupt-driven I/O lets the CPU do other work until the device requests service. That removes continuous status polling, which is the principal advantage in B. Service still consumes CPU time and needs device support/software. Therefore it does not universally increase device transfer speed, eliminate hardware requirements or remove drivers.

Why each choice is right or wrong

A - Incorrect: Avoiding polling does not by itself guarantee that the device or bus transfers data at a higher rate.

B - Correct: The CPU can execute other useful work between setup and the device's service request.

C - Incorrect: Interrupt notification requires suitable device/controller support; the note does not promise no special hardware.

D - Incorrect: A handler/driver must still service the device; interrupt-driven operation does not remove that software role.

Primary note proof

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA29

> A controller is hardware; a driver is software that operates it.

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Question sources

Computer Organization & Architecture review questions · Question 39

Question 174 | NS0406

Direct Memory Access (DMA) primarily improves performance by:

A. Increasing CPU clock speed

B. Allowing a controller to transfer data between an I/O device and memory without the CPU copying each word

C. Compressing data before transfer

D. Using a faster bus

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer B: Allowing a controller to transfer data between an I/O device and memory without the CPU copying each word

DMA transfers a block directly between an I/O device and memory after CPU setup. Its performance benefit is reduced per-word CPU copying, leaving the processor available for other work while the controller performs the transfer. CPU configuration and completion handling remain necessary, and DMA still consumes memory bandwidth. It does not inherently raise the clock, compress data or require a faster bus.

Why each choice is right or wrong

A - Incorrect: DMA changes who performs transfer work, not the processor's clock frequency.

B - Correct: The controller moves the block after setup, avoiding a CPU copy operation for every transferred word.

C - Incorrect: Data compression is not part of the DMA definition or its stated per-word CPU-work reduction.

D - Incorrect: DMA may use the existing bus and does not inherently replace it with a faster one.

Primary note proof

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA30

> DMA does not eliminate CPU involvement in configuration, ownership, address mapping or completion.

Study-note section COA30

> DMA reduces per-word CPU work but competes for memory bandwidth.

Question sources

Computer Organization & Architecture review questions · Question 40

Question 175 | NS0407

In Memory-Mapped I/O, how are I/O devices accessed?

A. Using special IN/OUT instructions

B. Through dedicated I/O ports separate from memory

C. By assigning memory addresses to I/O registers, accessed like regular memory

D. Via DMA exclusively

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Device registers occupy memory addresses and use suitable memory access instructions.

COA31 says memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable loads/stores. C matches, but like regular memory refers to address/instruction access rather than identical side effects: MMIO widths, ordering and status-read behavior still matter. Special IN/OUT and a separate port space instead describe isolated I/O; DMA is independent of address-space choice.

Why each choice is right or wrong

A - Incorrect: Special IN/OUT instructions describe isolated/port-mapped I/O rather than the defining memory-mapped access mechanism.

B - Incorrect: A separate port space is the isolated I/O arrangement, not device registers in memory address space.

C - Correct: MMIO places device registers at memory-space addresses accessed through suitable loads/stores, with device-specific side-effect rules.

D - Incorrect: MMIO specifies addressing and access, not an exclusive requirement to transfer all device data through DMA.

Primary note proof

Study-note section COA31

> Memory-mapped I/O assigns device registers addresses in the memory address space and uses suitable load/store operations.

Study-note section COA31

> Isolated/port-mapped I/O has a separate port space and special instructions such as IN/OUT.

Study-note section COA31

> MMIO must respect device access widths, ordering and side effects; a status read is not necessarily a harmless RAM read.

Question sources

Computer Organization & Architecture review questions · Question 41

Question 176 | NS0408

An I/O 'Write Command' instructs the I/O module to:

A. Read data from the peripheral and store it in a buffer

B. Check the device status for errors

C. Take data from the data bus and transmit it to the peripheral

D. Reset the peripheral device

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: Transmit system data to the peripheral.

COA29 defines write/output commands as sending system data to the device. C matches that direction: data from the system bus is handed to the peripheral. Reading brings device data toward the system, status testing inspects readiness/errors, and reset is a control action. These are distinct command functions even when one controller implements them all.

Why each choice is right or wrong

A - Incorrect: Moving peripheral data into a system buffer is the input/read direction rather than write/output.

B - Incorrect: Checking readiness or errors is a test/status command rather than transferring output data.

C - Correct: A write/output command sends data from the system side to the peripheral device.

D - Incorrect: Reset is a device-control action, not the ordinary data-output direction defining a write command.

Primary note proof

Study-note section COA29

> Control commands start actions, test/status commands inspect readiness or errors, read commands bring device data toward the system, and write/output commands send system data to the device.

Question sources

Computer Organization & Architecture review questions · Question 42

Question 177 | NS0409

Which I/O technique is most efficient for transferring large blocks of data, such as disk reads?

A. Programmed I/O

B. Interrupt-Driven I/O

C. Direct Memory Access (DMA)

D. Memory-Mapped I/O

Hint

Follow the data path and control responsibility; distinguish the transfer mechanism, address space and interface type.

Answer C: DMA for substantial block transfers.

COA30 explicitly recommends DMA for substantial blocks because it reduces per-word CPU work to setup and completion duties. This matches disk-like large transfers. Polling and interrupt transfer still involve CPU servicing of individual items, while memory-mapped I/O is an address-space design choice that may coexist with DMA. The answer assumes the substantial-block context, not a universal fastest mechanism for every tiny transfer.

Why each choice is right or wrong

A - Incorrect: Programmed I/O makes the CPU check status and move each item, adding substantial work for large blocks.

B - Incorrect: Interrupt-driven transfer avoids continuous polling but still consumes CPU service work for transferred items.

C - Correct: DMA replaces much per-item CPU work with block setup/completion, matching the substantial-transfer recommendation.

D - Incorrect: Memory mapping specifies device addressing and can coexist with DMA; it does not itself offload block movement.

Primary note proof

Study-note section COA30

> Programmed/polled I/O repeatedly checks status and lets the CPU move each item.

Study-note section COA30

> Interrupt-driven I/O lets the CPU do other work until an event requests service; the handler still consumes CPU time to service the device.

Study-note section COA30

> DMA lets a controller move a block directly between device and memory after CPU setup.

Study-note section COA30

> Choose it for substantial blocks; mapping devices to addresses is a separate design dimension.

Question sources

Computer Organization & Architecture review questions · Question 43

Question 178 | NS0410

SMP (Symmetric Multiprocessing) systems are characterized by:

A. Multiple different processors with specialized roles

B. Compatible peer processors sharing memory and I/O under one operating system

C. Processors with separate private address spaces communicating by messages

D. Independent networked computers without shared memory

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: Compatible peer processors sharing memory and I/O under one operating system

SMP gives processors peer roles with shared memory and I/O under an operating system. Compatibility and symmetric roles matter; every physical processor property need not be identical. Specialized asymmetric roles do not state that model, while separate address spaces and independent networked computers describe distributed organization. The alternatives compare coherent organizations without an impossible single-processor multi-chip description.

Why each choice is right or wrong

A - Incorrect: Specialized unequal processor roles describe an asymmetric organization rather than the peer-role model.

B - Correct: Peer roles with shared memory and I/O under one OS match the SMP definition.

C - Incorrect: Separate private address spaces and message communication describe distributed-memory organization.

D - Incorrect: Independent computers without shared memory do not have the stated SMP shared-memory organization.

Primary note proof

Study-note section COA37

> SMP processors share memory/I/O and have peer roles under an OS.

Study-note section COA37

> Distributed-memory systems communicate between separate address spaces.

Question sources

Computer Organization & Architecture review questions · Question 44

Question 179 | NS0411

Pipelining improves CPU performance by:

A. Increasing the clock speed of each stage

B. Executing one instruction at a time faster

C. Overlapping the execution stages of multiple instructions simultaneously

D. Using larger registers

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer C: Overlap stages of different instructions.

COA35 defines pipelining by overlapping stages from different instructions and says it primarily improves throughput. C directly states that mechanism. It does not require faster stages, larger registers or a single instruction completing all work faster. Pipeline fill/drain, stalls and the slowest stage constrain actual gains, so overlap rather than guaranteed latency reduction is the supported explanation.

Why each choice is right or wrong

A - Incorrect: Pipeline performance does not require increasing each stage's clock speed; clock choice is limited by the slowest stage.

B - Incorrect: An instruction still spans several stages, so lower single-instruction latency is not the defining improvement.

C - Correct: Overlapping stages lets different instructions progress concurrently, increasing completed-work throughput.

D - Incorrect: Larger register width does not define stage overlap or directly explain pipelining's throughput benefit.

Primary note proof

Study-note section COA35

> A pipeline overlaps stages of different instructions; it primarily improves throughput.

Study-note section COA35

> The ideal steady-state completion rate is one instruction per cycle, while an individual instruction still spans multiple stages.

Question sources

Computer Organization & Architecture review questions · Question 45

Question 180 | NS0412

In an arithmetic pipeline for floating-point addition, which is the FIRST sub-operation performed?

A. Normalize the result

B. Add/subtract the mantissas

C. Align the mantissas

D. Compare the exponents

Hint

Keep the radix, bit width and representation fixed; evaluate place values or arithmetic steps in order.

Answer D: Compare exponents first.

COA10 orders floating addition as compare exponents, shift the smaller significand to align, add/subtract, normalize and round. Comparing exponents must precede alignment because their difference determines the required shift. Adding before alignment combines unequal place weights, and normalization happens after arithmetic. Therefore D is the first named sub-operation.

Why each choice is right or wrong

A - Incorrect: Normalization follows aligned addition/subtraction rather than initiating the floating-point addition sequence.

B - Incorrect: Significands must first be aligned to a common exponent before their values can be added correctly.

C - Incorrect: Alignment follows exponent comparison, which determines which significand to shift and by how much.

D - Correct: Exponent comparison is explicitly first and supplies the information needed for subsequent alignment.

Primary note proof

Study-note section COA10

> For floating addition, compare exponents, shift the smaller significand to align, add or subtract, normalize and round.

Question sources

Computer Organization & Architecture review questions · Question 46

Question 181 | NS0413

Which statement describes vector-instruction semantics rather than guaranteeing simultaneous execution of every array element?

A. Processes one element at a time faster

B. One vector instruction applies an operation across multiple data elements, possibly in lane-sized batches

C. Uses fewer transistors

D. Operates at a higher clock frequency

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: One vector instruction applies an operation across multiple data elements, possibly in lane-sized batches

A vector instruction describes an operation over a sequence of elements. Its semantics cover multiple data elements, but physical lanes can process those elements in batches instead of executing the entire array at one instant. The choice distinguishes vector from scalar processing without requiring simultaneous execution of every element. Faster individual elements, fewer transistors or a higher clock are implementation properties rather than definitions of vector processing.

Why each choice is right or wrong

A - Incorrect: Faster scalar element processing does not define vector semantics.

B - Correct: This states the multi-element instruction semantics while permitting physical batching.

C - Incorrect: A transistor-count reduction is not part of the definition.

D - Incorrect: A higher clock frequency is not required by vector semantics.

Primary note proof

Study-note section COA37

> A vector instruction describes an operation over a sequence of elements; physical lanes may process them in batches, not all in one instant.

Question sources

Computer Organization & Architecture review questions · Question 47

Question 182 | NS0414

For an ideal single-issue RISC pipeline in steady state after filling, with balanced stages and no stalls, what is the completion throughput?

A. One instruction per two clock cycles

B. One instruction per clock cycle

C. Two instructions per clock cycle

D. Depends on instruction complexity

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: One instruction per clock cycle

An ideal single-issue pipeline overlaps stages of successive instructions. After filling, it completes one instruction each cycle when stages are balanced and no hazards or other stalls interfere. An individual instruction still crosses several stages, so this is throughput rather than one-cycle instruction latency. Specifying single issue excludes a superscalar interpretation that could complete more than one instruction per cycle.

Why each choice is right or wrong

A - Incorrect: One completion per two cycles is below the stated ideal no-stall rate.

B - Correct: The ideal steady-state single-issue completion rate is one per cycle.

C - Incorrect: Two per cycle requires multiple-issue resources beyond the stated design.

D - Incorrect: The ideal rate follows from the specified single-issue pipeline assumptions.

Primary note proof

Study-note section COA35

> In an ideal k-stage single-issue pipeline, n instructions take k+n−1 cycles when stages are balanced and there are no stalls.

Study-note section COA35

> The ideal steady-state completion rate is one instruction per cycle, while an individual instruction still spans multiple stages.

Question sources

Computer Organization & Architecture review questions · Question 48

Question 183 | NS0415

Flynn's taxonomy classifies parallel computer architectures based on:

A. Number of processors and memory size

B. Instruction streams and data streams

C. Clock speed and bus width

D. Cache size and register count

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer B: Instruction streams and data streams.

COA37 explicitly defines Flynn classification by instruction and data streams. SISD has one of each, SIMD one instruction stream with multiple data elements, and MIMD multiple streams of each. Processor count, capacity, clock and register/cache sizes are implementation properties rather than the two classification axes. Therefore B directly matches the taxonomy.

Why each choice is right or wrong

A - Incorrect: Processor count and memory size do not define the instruction-stream/data-stream axes of Flynn’s taxonomy.

B - Correct: Flynn classifies the number of instruction streams and data streams, producing SISD/SIMD/MISD/MIMD categories.

C - Incorrect: Clock and bus width measure implementation characteristics rather than Flynn’s stream-count classification.

D - Incorrect: Cache capacity and register count are not the two axes used by Flynn’s taxonomy.

Primary note proof

Study-note section COA37

> Flynn classifies instruction and data streams: SISD has one of each; SIMD applies one instruction stream to multiple data elements; MIMD uses multiple instruction and data streams; MISD has multiple instruction streams for a common data stream and is uncommon.

Question sources

Computer Organization & Architecture review questions · Question 49

Question 184 | NS0416

The primary goal of parallel processing is to increase:

A. Memory capacity

B. Clock frequency

C. Throughput - the amount of processing per unit time

D. Register size

Hint

Separate throughput from latency and check which operations overlap or share resources under the stated model.

Answer C: Throughput: completed processing per unit time.

COA37 says more cores increase available parallel work while serial dependencies and coordination limit speedup. COA34 defines throughput as completed work per unit time. Parallel processing aims to raise that work rate, matching C, rather than inherently increasing physical memory, clock frequency or register width. Actual gains still depend on enough independent work and manageable overhead.

Why each choice is right or wrong

A - Incorrect: Parallel execution does not inherently increase the system's installed memory capacity.

B - Incorrect: Additional concurrent work can improve throughput without increasing any processor's clock frequency.

C - Correct: Concurrent useful work aims to increase completed processing per unit time, subject to serial and coordination limits.

D - Incorrect: Register width is an architectural data-size property, not the primary objective of parallel execution.

Primary note proof

Study-note section COA37

> More cores increase available parallel work, but serial dependencies and coordination limit speedup.

Study-note section COA34

> Throughput counts completed work per unit time; latency measures one job's delay.

Question sources

Computer Organization & Architecture review questions · Question 50

Data Communication and Computer Networking

Questions & Worked Answers

274 questions + 1 source variants

Question 1 | Q00016

Which term describes combining multiple analogue or digital input signals for transmission over one shared medium?

A. Point-to-point

B. Broadcasting

C. Synchronization

D. Multiplexing

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer D: Multiplexing

Multiplexing combines separate input streams so they can share a transmission link. The receiver uses demultiplexing to separate them again. This operation concerns sharing link capacity, whereas broadcasting concerns the recipients of a message. Point-to-point describes the endpoints of a link, and synchronization coordinates timing. The option supplies the required operation while preserving those distinctions.

Why each choice is right or wrong

A - Incorrect: Point-to-point joins two endpoints and does not mean combining input signals onto a shared link.

B - Incorrect: Broadcast sends a message to all destinations in scope rather than combining multiple input streams.

C - Incorrect: Synchronization coordinates timing; it is not the multiplexing operation defined by the stem.

D - Correct: Multiplexing combines multiple streams for a shared link and is precisely the operation described.

Primary note proof

Study-note section NET32

> Multiplexing combines multiple input streams onto a shared link; demultiplexing separates them at the receiver. It differs from broadcasting one message to many destinations. FDM

assigns distinct frequency bands with suitable guard spacing; WDM assigns different optical wavelengths. TDM allocates time slots: synchronous TDM reserves recurrent slots even if an input is idle, while statistical TDM allocates according to demand and needs identification.

Study-note section NET08

Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Study-note section NET06

Simplex is one way, half duplex alternates directions, and full duplex permits simultaneous sending/receiving. A walkie-talkie is half duplex; a telephone conversation uses full duplex. Serial sends sequential bits; parallel uses multiple conductors and may suffer skew. Character-oriented asynchronous serial adds start/stop bits; synchronous framing uses ongoing clock recovery. Neither is synonymous with browser asynchronous requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 37-42

MULTIPLEXING
Whenever the bandwidth of a medium linking two devices is greater than the bandwidth needs of the devices, the link can be shared. Multiplexing is the set of techniques that allows the simultaneous transmission of multiple signals across a single data link.

Data Communication and Computer Networking module | PDF page 11 | lines 34-51

NETWORK TOPOLOGY
A network topology is a description of the layout of the region or area covered by that network.
There are two types of connections. They are: point-to-point and multi-point.
Point-to-point connections provide a direct link between two devices.
Multi-point connections provide a link between three or more devices on a network.
There are seven basic topologies in the study of network topology:
• Point-to-point topology,
• Bus (point-to-multipoint) topology,
• Ring topology,
• Star topology,
• Hybrid topology,
• Mesh topology and,
• Tree topology.

Data Communication and Computer Networking module | PDF page 17 | lines 54-59

> ICMP- Internet Control Message Protocol: It is a signaling mechanism used to inform the sender about datagram problemsthat occur during transit. It is used by intermediate devices.
> IGMP- Internet Group Message Protocol: It is a mechanism that allows sending the same message to a group of recipients.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 16

Question 2 | Q00036

A traditional phone call is an example of

A. Packet switching

B. Multiplexing

C. Broadcasting

D. Circuit switching

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: Circuit switching

A traditional circuit telephone call exemplifies establishment, continuous use of a dedicated path, and release. This applies to the traditional circuit service assumed by the stem, rather than all modern telephone implementations. Evidence: PDF 20, 20, 20.

Why each choice is right or wrong

A - Incorrect: Packets are discrete rather than the dedicated circuit described (PDF 20, 20, 20.)

B - Incorrect: Multiplexing shares a link and may underlie telephony but does not name its switching service (PDF 20, 20, 20.)

C - Incorrect: Broadcasting does not establish a dedicated pairwise call (PDF 20, 20, 20.)

D - Correct: Dedicated establishment, transfer and disconnect match the call (PDF 20, 20, 20.)

Primary note proof

Study-note section NET31

> Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not

make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Study-note section NET32

Multiplexing combines multiple input streams onto a shared link; demultiplexing separates them at the receiver. It differs from broadcasting one message to many destinations. FDM assigns distinct frequency bands with suitable guard spacing; WDM assigns different optical wavelengths. TDM allocates time slots: synchronous TDM reserves recurrent slots even if an input is idle, while statistical TDM allocates according to demand and needs identification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 12-26

Circuit Switching: In circuit switching, two communicating stations are connected by a dedicated communication path which consists of intermediate nodes in the network and the links that connect these nodes.
It involved the following three distinct steps
• Circuit Establishment: To establish an end-to-end connection before any transfer of data.
• Data transfer: Transfer data is from the source to the destination. The data may be analog or digital, depending on the nature of the network. The connection is generally full-duplex.
• Circuit disconnect: Terminate connection at the end of data transfer. Signals must be propagated to deallocate the dedicated resources.

Data Communication and Computer Networking module | PDF page 20 | lines 27-36

Message Switching:
Packet Switching: Packet switching was designed to address the shortcomings of circuit switching in dealing with data communication. Unlike circuit switching where communication is continuous along a dedicated circuit, in packet switching, communication is discrete in form of packets.
The three switching modes to choose from are:
• Cut-through
• Store-and-forward
• Fragment-free

Data Communication and Computer Networking module | PDF page 20 | lines 37-42

MULTIPLEXING
Whenever the bandwidth of a medium linking two devices is greater than the bandwidth needs of the devices, the link can be shared. Multiplexing is the set of techniques that allows the

simultaneous transmission of multiple signals across a single data link.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 36

Question 3 | Q00050

Assume you are searching for information and typed the domain name: www.mysearchexample.com on your browser, having an IP address 192.108.110.1. Since internet communication is based on an IP address, the domain name has to be translated to its corresponding IP address. Which of the following is responsible for the translation?

A. DNS

B. SMTP

C. FTP

D. DHCP

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer A: DNS

The needed operation is name resolution. The module identifies DNS as Domain Name Service and contrasts external DNS naming with locally stored host addresses; together these support the name-to-address lookup. Evidence: PDF 18, 65, 18.

Why each choice is right or wrong

A - Correct: DNS is the named external naming service (PDF 18, 65, 18.)

B - Incorrect: SMTP is electronic mail, not name lookup (PDF 18, 65, 18.)

C - Incorrect: FTP is file transfer, not name lookup (PDF 18, 65, 18.)

D - Incorrect: Page28 explicitly expands DHCP as Dynamic Host Configuration Protocol, separately from DNS (Domain Name Service). The external naming service discussed on page65 is DNS, the operation requested, rather than the separately named host-configuration protocol. The expansion alone is not a full account of DHCP operation.

Primary note proof

Study-note section NET20

DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

/etc/hosts
The /etc/hosts file always contains the localhost IP
address, 127.0.0.1, which is
usedfor interprocess communication.
Never remove this line! Sometimes contains
addresses of additional hosts, whichcan be contacted
without using an external naming service such as
DNS (the Domain Name Server).

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable
connection-oriented, reliable protocol. i.e. a connection is
established between the sender and receiver before the data can
be transmitted. It divides the data it receives from the upper
layer into segments and tags a sequence number to each
segment which is used at the receiving end for reordering of
data.
UDP - User Datagram Protocol: UDP is a simple protocol
used for process to process transmission. It is an unreliable,
connectionless protocol for applications that do not require flow
control or error control. It simply adds port address, checksum
and length information to the data it receives from the upper
layer.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

The Process/Application Layer Protocols
• Telnet

- File Transfer Protocol (FTP)
- Network File System (NFS).
- Simple Mail Transfer Protocol (SMTP
- POP3 is used to receive mail.
- Simple Network Management Protocol (SNMP)
- Domain Name Service (DNS)
- Dynamic Host Configuration Protocol (DHCP)

Question sources

MOE Computer Science Exit Exam · 2016 · Question 50

Question 4 | Q00061

Which one of the following is not correct about networking models?

A. TCP/IP model is a protocol model

B. A reference model has no implementation specification

C. A network designer can use either of TCP/IP or OSL, or both

D. Open system interconnection (OSI) is a protocol model

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: OSI is a reference model, not itself an implemented protocol suite

NET03 distinguishes an abstract reference model from an implemented protocol suite. D incorrectly calls OSI a protocol model in that distinction. TCP/IP supplies the implemented suite, and designers can use models as architectural guides. B is valid when understood as the model itself not providing executable implementation details, rather than denying all interface descriptions.

Why each choice is right or wrong

A - Incorrect: TCP/IP describes an implemented protocol suite, consistent with calling it a protocol model.

B - Incorrect: A reference model divides responsibilities/interfaces without itself being executable implementation software.

C - Incorrect: Models can guide network design; using one or both perspectives is compatible with their reference purpose.

D - Correct: OSI is the reference model in the stated contrast, not itself the implemented TCP/IP-style protocol suite.

Primary note proof

Study-note section NET03

A reference model divides responsibilities and interfaces; it is not itself executable networking software. OSI is a seven-layer reference model; TCP/IP describes an implemented protocol suite, usually taught with four layers. A five-layer teaching model separates physical and link functions. State the model before counting layers.

Study-note section NET19

The four-layer TCP/IP model folds OSI layers 5-7 into application, retains transport, calls layer 3 Internet, and combines link/physical as network access. A model is a guide to functions, not a guarantee that every application gets a reliable network. Distinguish end-to-end application service from its underlying process-to-process transport.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 12-29

The Open Systems Interconnection (OSI) Model was
developed by International Organization for Standardization
(ISO). ISO is the organization, OSI is the model
It was developed to allow systems with different platforms
(Hardware & software) to communicate with each other.
Now, it is a network model that defines the protocols for
network communications.
It is a hierarchical model that groups its processes into seven
(7) layers. (Top to Bottom)
1. Application Layer
2. Presentation Layer
3. Session Layer
4. Transport Layer
5. Network Layer
6. Data Link Layer
7. Physical Layer
Each layer has specific functions to perform and has to co-
operate with the layers above and below it.

Data Communication and Computer Networking module | PDF page 16 | lines 50-59

After understanding the concept of layered task and the OSI
model, we introduce the TCP/IP model. This model is
currently being used on our systems. TCP/IP model is a
collection of protocols often called a protocol suite. It offers a
rich variety of protocols.
It is a hierarchical model; it existed even before the OSI model
was developed. Originally had four layers (bottom to top):
1. Host to Network Layer
2. Internet Layer
3. Transport Layer

Data Communication and Computer Networking module | PDF page 25 | lines 17-27

TCP/IP is a hierarchical protocol made up of interactive modules, each of which provides a specific functionality; however, the modules are not necessarily interdependent. Whereas the OSI model specifies which functions belong to each of its layers, the layers of TCP/IP suite contain relatively independent protocols that can be mixed and matched depending on the needs of the system. The term hierarchical means that each upper-level protocol is supported by one or more lower-level protocols.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 61

Question 5 | Q00063

Which statement about IPv4 is incorrect?

A. It uses 128-bit addresses.

B. Its finite address space has faced allocation scarcity.

C. Private addressing with NAT can conserve public IPv4 addresses.

D. It uses 32-bit addresses.

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer A: It uses 128-bit addresses.

IPv4 addresses contain 32 bits, whereas IPv6 uses a much larger address space. IPv4 allocation scarcity motivates conservation through private addressing and NAT. These are address-space properties, not a claim that every IPv4 router is intrinsically slow or memory-intensive. The incorrect statement assigns a 128-bit address width to IPv4.

Why each choice is right or wrong

A - Correct: IPv4 addresses are 32 bits, so the claimed 128-bit width is incorrect.

B - Incorrect: Finite IPv4 address allocation has faced scarcity; this statement is true.

C - Incorrect: Private addressing with NAT conserves public IPv4 addresses; this statement is true.

D - Incorrect: The stated 32-bit IPv4 address width is correct.

Primary note proof

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Study-note section NET-S03

The finite IPv4 address space has faced allocation scarcity. Private addressing with NAT conserves public IPv4 addresses, while IPv6 provides a much larger address space.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the Internet. Every host and router on the Internet has an IP address, which encodes its network number and host number.
• The combination is unique: in principle, no two machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination address fields of IP packets.
• An IP address does not refer to a host but it refers to a network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of network addresses in IPv4 led to the development of IPv6. Due to this, we need to use private IP address inside the organization and translate it to public IP address using NAT.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 63

Question 6 | Q00073

Which item is not a transmission impairment?

A. Synchronization

B. Crosstalk

C. Impulse noise

D. Distortion

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer A: Synchronization

Distortion changes the signal's shape, while crosstalk and impulse noise add unwanted disturbances. Crosstalk couples energy from a nearby channel; impulse noise is a brief disturbance. Synchronization instead aligns timing so the receiver can interpret transmitted symbols. Losing synchronization can cause errors, but synchronization itself is a desirable coordination process rather than one of these signal impairments.

Why each choice is right or wrong

A - Correct: Synchronization aligns timing and is not itself an impairment.

B - Incorrect: Crosstalk is unwanted coupling from nearby channels, a noise impairment.

C - Incorrect: Impulse noise is a short unwanted disturbance, a noise impairment.

D - Incorrect: Distortion changes the signal shape and is a transmission impairment.

Primary note proof

Study-note section NET06

> Attenuation reduces signal strength. Distortion changes shape, often because frequencies have different delays. Noise adds unwanted energy: thermal noise comes from random thermal motion, crosstalk leaks from nearby channels, and impulse noise is a short disturbance. Amplifiers amplify noise too; digital repeaters regenerate decisions but cannot reconstruct every corrupted bit.

Study-note section NET06

> Simplex is one way, half duplex alternates directions, and full duplex permits simultaneous sending/receiving. A walkie-talkie is half duplex; a telephone conversation uses full duplex. Serial sends sequential bits; parallel uses multiple conductors and may suffer skew. Character-oriented asynchronous serial adds start/stop bits; synchronous framing uses ongoing clock recovery. Neither is synonymous with browser asynchronous requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 10 | lines 18-35

> receiver.There are three causes of impairment:
> • Attenuation
> • Distortion
> • Noise
> Attenuation
> Attenuation results in loss of energy due to distance.
> Whena signal travels through a medium, it loses some
> of its energy in overcoming the resistance of the

medium. The electrical energy in the signal may convert to heat. To compensate for this loss, amplifiers are used to amplify thesignal.
Distortion
Distortion changes the shape of the signal.
Noise
Noise is any unwanted signal that is mixed or combined with the original signal during transmission. Due to noise the original signal is altered and signal received is not sameas the one sent.

Data Communication and Computer Networking module | PDF page 14 | lines 50-55

• Synchronization of bits: To interpret correct and accurate data the sender and receiver have to maintain the same bit rate and also have synchronized clocks.
• Line configuration: The physical layer defines the nature of the connection .i.e. a point to point link, or a multi-point link.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 73

Question 7 | Q00086

_____ is a probabilistic algorithm where a data packet is sent by the router to any one of its neighbors randomly.

A. Adaptive routing algorithm

B. Flooding routing algorithm

C. Random walks routing algorithm

D. Distributed routing algorithm

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer C: Random-walk routing

NET30 defines a random walk by probabilistically choosing a neighbor, exactly the forwarding behavior in the stem. Flooding instead replicates packets. Adaptive and distributed describe broader properties of routing algorithms; neither alone specifies random single-neighbor selection.

Why each choice is right or wrong

A - Incorrect: Adaptive routing responds to network conditions; adaptation alone does not mean choosing a random neighbor.

B - Incorrect: Flooding sends copies on multiple eligible links rather than selecting just one neighbor randomly.

C - Correct: Random walk selects the next neighbor probabilistically, matching the stated packet-forwarding procedure.

D - Incorrect: Distributed routing spreads computation among routers; it does not specify the random choice described.

Primary note proof

Study-note section NET30

Flooding sends copies on eligible outgoing links with duplicate/lifetime controls; random walk picks a neighbor probabilistically. Unicast targets one destination, broadcast all in a scope, multicast a subscribed group and anycast one member sharing an address, normally nearest under routing metrics. Multi-destination packets can list multiple targets; this is not necessarily identical to broadcast. Group multicast trees use mechanisms such as reverse-path checks; Ethernet spanning tree is a different layer-2 mechanism.

Study-note section NET-S01

Adaptive or dynamic routing changes routes in response to updated network information; nonadaptive or static routing uses configured choices. Distributed routing means routers perform computations using exchanged information instead of requiring one central route computation. These properties do not imply random next-hop selection: the path-selection rule is a separate classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 52-64

ROUTING AND ROUTING PROTOCOLS
Routing is the act of moving information across an internetwork from a source to a destination. At the heart of such protocol is the routing algorithm that determines the path for a packet.
• Routing requires a host or a router to have a routingtable which is constructed by the routing algorithm.
• There are wo kinds of routing algorithms (routing tables): nonadaptive (static) and adaptive (dynamic)
• And two kinds of protocols
o Interior routing protocols
o Exterior routing protocols

Question sources

MOE Computer Science Exit Exam · 2016 · Question 86

Question 8 | Q00087

Which network address translation protocol solves the task of address resolution using port number?

A. Network Address Translation

B. Port Address Translation

C. Dynamic Address Translation

D. Static address translation

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: Port Address Translation (PAT/NAPT)

The wording loosely calls translation “address resolution,” but its decisive qualifier is use of port numbers. NET29 identifies PAT/NAPT as translation of address-plus-port combinations, allowing shared public addresses. This is not hardware-address resolution by ARP, and ordinary static or pooled dynamic address mappings do not themselves describe port multiplexing.

Why each choice is right or wrong

A - Incorrect: NAT is the broad translation family; this option does not identify the port-based form specifically.

B - Correct: PAT translates address-and-port combinations, distinguishing simultaneous clients that share an external address.

C - Incorrect: Dynamic NAT allocates address mappings from a pool, without necessarily multiplexing transport ports.

D - Incorrect: Static NAT fixes an address mapping; port-based sharing is the separate PAT mechanism.

Primary note proof

Study-note section NET29

> Static NAT gives a fixed address mapping; dynamic NAT allocates mappings from a pool. PAT/NAPT maps address-plus-port combinations so multiple private clients share a public address. It is translation, not ARP resolution and not encryption. Return traffic must match the translation state or a configured inbound mapping.

Study-note section NET20

> ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target’s MAC. It

bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of network addresses in IPv4 led to the development of IPv6. Due to this, we need to use private IP address inside the organization and translate it to public IP address using NAT.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Data Communication and Computer Networking module | PDF page 28 | lines 2-14

TCP and UDP must use port numbers to communicate with the upper layers, because they‘re what keep track of different conversations crossing the network simultaneously. These port numbers identify the source and destination application or process in the TCP segment. There are 216 = 65,536 ports available.
Well-known ports: The port numbers range from 0 to 1,023.
Registered ports: The port numbers range from 1,024 to 49,151. Registered ports are used by applications or services that need to have consistent port assignments.
Dynamic or private ports: The port numbers range from 49,152 to 65,535.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 87

Question 9 | Q00088

Which protocol resolves a known on-link IPv4 next-hop address to its link-layer hardware address?

A. Address Resolution Protocol (ARP)

B. Reverse Address Resolution Protocol (RARP)

C. Bootstrap Protocol (BOOTP)

D. Internet Control Message Protocol (ICMP)

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer A: Address Resolution Protocol (ARP)

ARP resolves an on-link IPv4 next hop to a link address, normally by broadcasting a request and receiving a reply. For an off-subnet destination, the next hop is usually the gateway rather than the remote destination itself. RARP reverses hardware-to-IP lookup; BOOTP supplies boot configuration, and ICMP carries network diagnostics. These distinct functions do not replace ARP's local IPv4-to-link resolution.

Why each choice is right or wrong

A - Correct: ARP resolves the known local IPv4 next hop to its link address.

B - Incorrect: RARP performs the reverse hardware-to-IP lookup.

C - Incorrect: BOOTP supplies boot configuration rather than resolving a known next-hop IPv4 address to its link address.

D - Incorrect: ICMP reports network errors and diagnostics rather than resolving hardware addresses.

Primary note proof

Study-note section NET20

> ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target's MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS

server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to
resolve thephysical address of a device on a network,
where its logical address is known.
Physical address is the 48 bit address that is imprinted
on theNIC or LAN card, Logical address is the Internet
Address orcommonly known as IP address that is used
to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is
used bya device on the network to find its Internet
address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 17 | lines 54-59

ICMP- Internet Control Message Protocol: It is a
signaling mechanism used to inform the sender about
datagram problemsthat occur during transit. It is used by
intermediate devices.
IGMP- Internet Group Message Protocol: It is a mechanism
that allows sending the same message to a group of recipients.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the
combined session, presentation, and application layers in
the OSI model and many protocols are defined at this
layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

MOE Computer Science Exit Exam · 2016 · Question 88

Question 10 | Q00089

A 172.16.0.0/16 parent network is divided into equal /19 broadcast subnets. Counting all subnets, which statement is correct?

A. It provides 8 subnets, 32 IP addresses each

B. It provides 8 subnets, 8,190 hosts each

C. It provides 8 subnets, 30 hosts each

D. It provides 8 subnets, 2,046 IP addresses each

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: It provides 8 subnets, 8,190 hosts each

Moving from /16 to /19 borrows three subnet bits, producing 2^3 = 8 equal subnets. Each /19 leaves thirteen host bits, so it contains 8192 addresses. Reserving the network and directed-broadcast addresses leaves 8190 usable hosts. Subnet counts describe subdivision of the stated parent; they must not be confused with addresses or hosts within a child subnet.

Why each choice is right or wrong

A - Incorrect: Eight subnets is correct, but each /19 has 8192 total addresses rather than 32.

B - Correct: Three borrowed bits yield eight subnets; thirteen host bits yield 8190 usable hosts per subnet.

C - Incorrect: Thirty usable hosts belongs to /27, not /19.

D - Incorrect: A /19 contains 8192 addresses, not 2046.

Primary note proof

Study-note section NET26

> To split parent /a into equal /p blocks, borrow b=p-a bits and obtain 2^b subnets. Do not subtract two subnets under modern practice. For a Class B /16 parent split to /19, b=3 gives eight subnets, each with 8192 addresses and 8190 conventional hosts. A /19 alone does not state its parent.

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

> IP addresses: IP address uniquely identifies a device on the
> Internet. Every host and router on the Internet has an IP address,

which encodes its network number and host number.
• The combination is unique: in principle, no two machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination address fields of IP packets.
• An IP address does not refer to a host but it refers to a network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address, the address space is 2N because each bit can have twodifferent values (0 or 1) and N bits can have 2N values.
• IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex.

255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 89

Question 11 | Q00090

A network device that provides the physical interface between computer and cabling is known as_____.

A. Switch

B. Router

C. NIC

D. Repeater

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer C: NIC

The star-topology description places RJ-45 cable connections at the NIC of each computer; the physical address is imprinted on that NIC. This identifies the host cabling interface. Evidence: PDF 23, 17, 12, 12, 11.

Why each choice is right or wrong

A - Incorrect: A switch forwards among attached devices (PDF 23, 17, 12, 12, 11.)

B - Incorrect: A router interconnects networks (PDF 23, 17, 12, 12, 11.)

C - Correct: The module names Network Interface Card and connects cable to it (PDF 23, 17, 12, 12, 11.)

D - Incorrect: A repeater regenerates a signal on a LAN (PDF 23, 17, 12, 12, 11.)

Primary note proof

Study-note section NET09

A NIC supplies a computer’s network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Study-note section NET10

A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the

incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET11

Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to resolve thephysical address of a device on a network, where its logical address is known.
Physical address is the 48 bit address that is imprinted on theNIC or LAN card, Logical address is the Internet Address orcommonly known as IP address that is used to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is used bya device on the network to find its Internet address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device used tointerconnect different types of network that uses different protocol.

Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to forwardpackets.
• The list of routing tables provides directions to transfer the data to a particular network destination -various paths to forward the data.
• A router analyzes a destination IP address of a givenpacket and compares it with the routing table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets across anetwork.
Unlike a hub which simply replicates what it receives on oneport onto all the other ports, a switch keeps a record of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the packetdirectly to the recipient device by looking up the MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 11 | lines 54-55

Repeater: A repeater is a device that operates only in the physical layer. A repeater receives a signal and, before it

Question sources

MOE Computer Science Exit Exam · 2016 · Question 90

Question 12 | Q00141

In the OSI teaching model, which layers provide hop-local or end-to-end flow control and error recovery when the chosen protocols support them? Select all that apply; isolated header-error detection alone does not qualify.

A. Application

B. Presentation

C. Network

D. Transport

E. Session

F. Data link

G. Physical

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D, F: Transport; Data link

Data-link protocols can provide hop-local flow control and recovery, such as acknowledgments and retransmission. Transport protocols such as TCP provide end-to-end flow control and reliable delivery. These capabilities depend on the protocol: Ethernet detection alone does not guarantee recovery, and UDP does not provide TCP reliability. IPv4 checks its header but remains best-effort and does not supply the requested end-to-end recovery and flow-control service.

Why each choice is right or wrong

A - Incorrect: Application supplies application-facing network services, not the specified OSI control role.

B - Incorrect: Presentation concerns representation, compression and encryption roles.

C - Incorrect: IPv4 header checks detect some errors, but IP lacks the requested recovery and flow-control service.

D - Correct: Transport protocols such as TCP provide end-to-end flow control and recovery.

E - Incorrect: Session manages dialog and checkpoints rather than the specified delivery controls.

F - Correct: Data-link protocols can provide hop-local flow control and recovery.

G - Incorrect: Physical signaling transmits symbols rather than implementing these delivery-control procedures.

Primary note proof

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits

distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET19

OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint

connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple

intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Question sources

Addis Ababa University mock exam · Question 43

Question 13 | Q00142

Which of the following statement(s) is/are WRONG? (Choose all that applies)

A. Internet/Network layer is responsible for routing

B. Physical address can be assigned by the network administrator.

C. Physical layer is concerned with specifying the characteristics of the transmission medium, the nature of the signals and the data rate.

D. Transport layer provides only unreliable data transmission.

E. Network layer is responsible for host to host communication.

F. None of the above

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: The transport layer does not provide only unreliable transmission

The question asks for WRONG statements. TCP is a transport-layer protocol with reliable ordered delivery, disproving the absolute “only unreliable” in D. The other substantive statements fit the layer model: network routing and host delivery, physical signaling specifications, and administratively assigned MAC addresses. NET33 explicitly permits locally administered MAC addresses, so B must not be rejected.

Why each choice is right or wrong

A - Incorrect: This statement is true: routing is a network/Internet-layer responsibility, so it is not a wrong statement.

B - Incorrect: NET33 permits locally administered MAC addresses; physical addresses need not always be factory assigned.

C - Incorrect: Physical signaling, transmission-medium characteristics and data rate belong to the physical layer.

D - Correct: The word only is false: TCP supplies reliable transport, whereas UDP lacks built-in delivery guarantees.

E - Incorrect: Host-to-host packet delivery is a network-layer function, distinct from transport process delivery.

F - Incorrect: At least D is wrong, so none of the above cannot answer the requested negative test.

Primary note proof

Study-note section NET33

> First identify the exact layer, protocol and scope. Flow/error control can exist at both link and transport layers; MAC addresses can be locally administered; bridges and switches both operate at layer 2. A LAN can use public or private unicast space. UDP and IP are both connectionless. Session dialogs and TCP connections are distinct senses of connection. Distinguish the protocol, layer and scope before treating these descriptions as exclusive.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET09

A NIC supplies a computer's network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.

• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to resolve thephysical address of a device on a network, where its logical address is known.
Physical address is the 48 bit address that is imprinted on theNIC or LAN card, Logical address is the Internet Address orcommonly known as IP address that is used to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is used bya device on the network to find its Internet address when it knows its physical address.

Question sources

Addis Ababa University mock exam · Question 44

Question 14 | Q00143

Which one of the following Medium Access Control protocols is supported by WLAN (802.11)?

A. ALOHA

B. CSMA/CA

C. Token Passing

D. FDMA

E. CSMA

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer B: CSMA/CA

NET14 names CSMA/CA as the 802.11 Wi-Fi access procedure. It combines carrier sensing, waiting/backoff and acknowledgments rather than Ethernet-style collision detection or token turns. Generic CSMA describes its family, but B gives the specifically supported WLAN mechanism requested by the stem; E omits its collision-avoidance procedure.

Why each choice is right or wrong

A - Incorrect: ALOHA sends opportunistically and is not the 802.11 contention mechanism identified in NET14.

B - Correct: CSMA/CA is the specifically identified WLAN mechanism, using waiting/backoff and acknowledgments.

C - Incorrect: Token passing is controlled access, unlike the contention procedure used by ordinary 802.11 WLANs.

D - Incorrect: FDMA divides frequency resources; it is not the stated WLAN medium-access protocol.

E - Incorrect: CSMA names the broader carrier-sensing family but omits the specific collision-avoidance WLAN protocol offered in B.

Primary note proof

Study-note section NET14

> Random access allows contention: ALOHA sends opportunistically; CSMA senses before sending. In legacy half-duplex Ethernet, CSMA/CD detects collision, sends a jam indication, stops, and retries after random backoff. The jam signals a collision, not its clearance. Wi-Fi 802.11 uses CSMA/CA with waiting/backoff and acknowledgments; sensing does not eliminate hidden-terminal collisions.

Study-note section NET14

Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 21 | lines 6-26

MEDIUM ACCESS CONTROL
The problem of controlling the access to the medium is similarto the rules of speaking in a meeting (the right to speak isupheld; two people do not speak at the same time; do notinterrupt each other; do not monopolize the discussion; ...) Three categories of multiple access protocols
Random Access Protocols - try your best like taxis do
• MA - Multiple Access
• CSMA - Carrier Sense MA
• CSMA/CD - CSMA with Collision Detection
• CSMA/CA - CSMA with Collision AvoidanceControlled-Access Protocols - get permission
• Reservation
• Polling
• Token Passing
Channelization Protocols - simultaneous use
• FDMA - Frequency-Division MA
• TDMA - Time-Division MA
• CDMA - Code-Division MA

Question sources

Addis Ababa University mock exam · Question 45

Question 15 | Q00144

Which device category can perform protocol conversion at higher OSI layers, potentially including application formats, rather than being limited to basic repeating, MAC bridging or IP routing?

A. Hub

B. Switch

C. Router

D. Gateway

E. Bridge

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Gateway

A protocol gateway translates between protocol or application representations, so its possible scope extends above basic link forwarding and IP routing. A hub repeats signals, a basic switch or bridge forwards using link-layer information, and a basic router forwards IP packets. Gateway is used in several senses: an ordinary default gateway is usually a router and need not implement conversion across every OSI layer.

Why each choice is right or wrong

A - Incorrect: A hub repeats physical signals rather than converting application protocols.

B - Incorrect: A basic layer-2 switch forwards frames by link-layer information.

C - Incorrect: A basic router chooses IP next hops rather than translating application formats.

D - Correct: A protocol-conversion gateway can translate higher-layer formats.

E - Incorrect: A bridge forwards between links using MAC information rather than higher-layer protocol conversion.

Primary note proof

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Study-note section NET09

> A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor

eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Study-note section NET10

A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is normallyused to create connections between stations in a physical startopology.
HUB: Hubs are commonly used to connect segments (devices)of a LAN. Hubs don‘t filter information; instead broadcast incoming packets to all computers connected to it. A hub contains multiple ports which support 8, 12 or 24 RJ-45 ports. Two types: Passive hub and Active hub. It is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets across anetwork.
Unlike a hub which simply replicates what it receives on oneport onto all the other ports, a switch keeps a record of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the packetdirectly to the recipient device by looking up the MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device used tointerconnect different types of network that uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to

forwardpackets.
• The list of routing tables provides directions to transfer the data to a particular network destination -various paths to forward the data.
• A router analyzes a destination IP address of a givenpacket and compares it with the routing table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 49 | lines 34-37

Gateways are basically protocol converters, facilitating compatibility between two protocols and operating on any layer of the open systems interconnection (OSI) model.

Data Communication and Computer Networking module | PDF page 12 | lines 60-61

BRIDGE: Connects two LANS having the same protocol - (e.g. Ethernet or Token ring).

Question sources

Addis Ababa University mock exam · Question 46

Question 16 | Q00145

Which listed address can be assigned as an RFC1918 private unicast host address on a correctly configured /24 LAN? Select one.

A. 127.0.0.0

B. 9.0.0.1

C. 192.168.1.256

D. 172.16.1.27

E. All of the above

F. None of the above

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer D: 172.16.1.27

An ordinary host address must have valid octets, belong to the required address scope and avoid the subnet network and broadcast addresses. The address 172.16.1.27 belongs to private 172.16.0.0/12 and is an interior host of 172.16.1.0/24. The public address 9.0.0.1 does not meet the private-address requirement. Loopback and an octet of 256 fail the other checks.

Why each choice is right or wrong

A - Incorrect: 127/8 is reserved for loopback and is not an ordinary LAN interface assignment.

B - Incorrect: 9.0.0.1 is public unicast, so it fails the explicit RFC1918 private-address requirement.

C - Incorrect: An IPv4 octet must be 0-255; 256 makes this address syntactically invalid.

D - Correct: 172.16.1.27 is within private 172.16/12 and is neither the network nor broadcast of its /24.

E - Incorrect: All fails because loopback and the impossible .256 octet are not valid ordinary LAN host assignments.

F - Incorrect: A private usable host is offered, so none of the above is false.

Primary note proof

Study-note section NET23

RFC1918 private blocks are 10.0.0.0/8, 172.16.0.0/12 (through 172.31.255.255), and 192.168.0.0/16. They can be reused internally but are not globally routed as public destinations. A LAN may also use legitimately assigned public addresses; private is common, not mandatory. 127.0.0.0/8 is IPv4 loopback, commonly 127.0.0.1, for local protocol access without an external link. 255.255.255.255 is limited broadcast; 0.0.0.0 can identify an unconfigured source. Link-local/APIPA uses 169.254.0.0/16, with edge /24 blocks excluded from normal automatic selection.

Study-note section NET25

For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Study-note section NET22

Write IPv4 as four decimal octets 0-255, or as 32 binary bits. Use weights 128,64,32,16,8,4,2,1 within each octet. Thus 11000001 10000011 00011011 11111111 is 193.131.27.255. A value such as 257 is impossible. Syntactic validity differs from assignability: an address ending .0 or .255 can be a host under some prefixes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).

Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of network addresses in IPv4 led to the development of IPv6. Due to this, we need to use private IP address inside the organization and translate it to public IP address using NAT.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

/etc/hosts
The /etc/hosts file always contains the localhost IP address, 127.0.0.1, which is usedfor interprocess communication.
Never remove this line! Sometimes contains addresses of additional hosts, whichcan be contacted without using an external naming service such as DNS (the Domain Name Server).

Question sources

Addis Ababa University mock exam · Question 47

Question 17 | Q00146

For the switching methods below, select all statements that are correct. In D, allocation on demand means reservation when a circuit is established.

A. Circuit Switching and Virtual Packet Switching methods reserve the required resources for the duration of the session during communication between end systems

B. Circuit switching technique uses store and forward method.

C. In message switching technique, it is not required to create a dedicated path.

D. In circuit switching techniques, resources are allocated on demand.

E. Packet switching can have nonuniform delay because packets share links and may queue.

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer C, D, E: In message switching technique, it is not required to create a dedicated path.; In circuit switching techniques, resources are allocated on demand.; Packet switching can have nonuniform delay because packets share links and may queue.

Message switching stores and forwards whole messages without a dedicated end-to-end path. Circuit switching instead reserves resources when a call is established and retains them through the session. Packet switching shares links, so queueing can produce variable delay, although packet voice remains possible. Virtual-circuit forwarding state alone does not require resource reservation. These distinctions support the three selected statements without equating all connection-oriented methods.

Why each choice is right or wrong

A - Incorrect: Circuit switching reserves resources, but virtual-circuit packet switching need not reserve them throughout a session.

B - Incorrect: Circuit switching uses an established reserved path; complete-message or packet buffering describes store-and-forward switching.

C - Correct: Message switching forwards stored messages without requiring a dedicated end-to-end circuit.

D - Correct: Circuit resources are allocated when a connection is requested and reserved for its session.

E - Correct: Shared links and queueing can give packets different delays.

Primary note proof

Study-note section NET31

> Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Study-note section NET31

> Datagram packets are forwarded independently. Virtual circuits establish forwarding state, but resource reservation is optional, not a necessary property of every virtual circuit. Message switching stores a complete message before forwarding, without a dedicated end-to-end path. Packet store-and-forward waits for the full packet/frame; cut-through starts after enough header arrives; fragment-free historically waits through the initial collision-sensitive portion. No routing algorithm guarantees that every packet always takes the geometrically shortest route.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 12-26

Circuit Switching: In circuit switching, two communicating stations are connected by a dedicated communication path which consists of intermediate nodes in the network and the links that connect these nodes.
It involved the following three distinct steps
• Circuit Establishment: To establish an end-to-end connection before any transfer of data.
• Data transfer: Transfer data is from the source to the destination. The data may be analog or digital, depending on the nature of the network. The connection is generally full-duplex.
• Circuit disconnect: Terminate connection at the end of data transfer. Signals must be propagated to deallocate the dedicated resources.

Data Communication and Computer Networking module | PDF page 20 | lines 27-36

Message Switching:
Packet Switching: Packet switching was designed to address the shortcomings of circuit switching in dealing with data communication. Unlike circuit switching where communication is continuous along a dedicated circuit, in packet switching, communication is discrete in form of packets.
The three switching modes to choose from are:
• Cut-through
• Store-and-forward
• Fragment-free

Question sources

Addis Ababa University mock exam · Question 48

Question 18 | Q00147

Which information is not correct about the IP address 172.16.35.72/27?

A. 172.16.35.64 is one of the subnet addresses.

B. The default mask for the network is 255.255.0.0.

C. Itis Class B address.

D. The number of bits used for the host address is 5.

E. The broadcast address of the second subnet is 172.16.35.255

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer E: The claimed second-subnet broadcast is wrong

For /27, five host bits give blocks of 32 in the last octet. The block containing .72 is .64-.95, with broadcast .95. If second subnet means the second /27 inside .35.0/24, its broadcast is .63; neither reading gives .255. The historical class is B and default mask /16, although the actual subnet mask is /27.

Why each choice is right or wrong

A - Incorrect: The .64 boundary is a valid /27 subnet start and contains host .72.

B - Incorrect: As a historical Class B address, its default mask is /16, even though the supplied actual prefix is /27.

C - Incorrect: First octet 172 is in the historical Class B range 128-191.

D - Incorrect: A 32-bit IPv4 address with /27 leaves exactly five bits for the host portion.

E - Correct: The second /27 in the local /24 broadcasts at .63; the containing subnet broadcasts at .95, never .255.

Primary note proof

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

> IP addresses: IP address uniquely identifies a device on the
> Internet. Every host and router on the Internet has an IP address,
> which encodes its network number and host number.
> • The combination is unique: in principle, no two
> machines on the Internet have the same IP address.
> • An IPv4 address is 32 bits long
> • They are used in the Source address and Destination
> address fields of IP packets.

• An IP address does not refer to a host but it refers to a network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address, the address space is 2N because each bit can have twodifferent values (0 or 1) and N bits can have 2N values.
• IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Addis Ababa University mock exam · Question 49

Question 19 | Q00156

Which one of the following is false about HTTP protocol?

A. Itis a stateless protocol

B. An HTTP message body can be empty

C. Itis a protocol used for communication between a web browser and a web server

D. None of the above

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer D: None of the listed HTTP claims is false

All three substantive choices agree with NET21: HTTP is stateless, a message body may be absent, and HTTP defines web request/response exchanges such as those between browsers and servers. The negative stem therefore has no false statement among A-C; D is the offered none-of-the-above response.

Why each choice is right or wrong

A - Incorrect: Statelessness is an HTTP property given in NET21, so this statement is true rather than false.

B - Incorrect: An HTTP message can omit its body, making this statement true.

C - Incorrect: Browser-to-web-server requests and responses are an ordinary use of HTTP, consistent with its web semantics.

D - Correct: A, B and C are all true, so none supplies the false statement requested.

Primary note proof

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET21

> Usual service ports: FTP control 21, SSH 22, Telnet 23, SMTP relay 25, DNS 53, HTTP 80, POP3 110, IMAP 143, HTTPS 443. Telnet provides unencrypted remote terminals; SSH protects remote access. Servers can be configured on other ports. A browser's HTTP exchange does not determine the physical duplex mode of its network interface.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 23-31

Telnet 23 SNMP 161
SMTP 25 TFTP 69
HTTP 80 DNS 53
FTP 21 POP3 110
DNS 53 DHCP 68
HTTPS 443 NTP 123
SSH 22 RPC 530
Table. Sample TCP and UDP port numbers from well-known category

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the combined session, presentation, and application layers in the OSI model and many protocols are defined at this layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Addis Ababa University mock exam · Question 58

Question 20 | Q00204

POP3 is most commonly used for:

A. Managing devices (such as routers and switches) on a network

B. Mapping IP addresses to hardware addresses

C. Sending emails from an email client to an email server

D. Retrieving emails from an email server to an email client

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: Retrieving emails from an email server to an email client

The module explicitly says POP3 is used to receive mail. Retrieval from a mail server into a client is that receiving function. Evidence: PDF 28, 17, 18.

Why each choice is right or wrong

A - Incorrect: Management is associated with SNMP, not POP3 (PDF 28, 17, 18.)

B - Incorrect: Address-to-hardware mapping is ARP (PDF 28, 17, 18.)

C - Incorrect: Sending mail is the SMTP role (PDF 28, 17, 18.)

D - Correct: Receiving mail matches the explicit POP3 statement (PDF 28, 17, 18.)

Primary note proof

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET-S02

SNMP means Simple Network Management Protocol. Its management applications monitor and control managed devices, such as hosts and routers, by exchanging management information with their agents. This device-management role differs from transferring email, resolving names or mapping an IP address to a MAC address.

Study-note section NET20

ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target's MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the
combined session, presentation, and application layers in
the OSI model and many protocols are defined at this
layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)

• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to resolve thephysical address of a device on a network, where its logical address is known.
Physical address is the 48 bit address that is imprinted on theNIC or LAN card, Logical address is the Internet Address orcommonly known as IP address that is used to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is used bya device on the network to find its Internet address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Mock exam 4 · Question 6

Question 21 | Q00205

The Address Resolution Protocol maps addresses between which layers?

A. Application layer to Network layer

B. Application layer to Transport layer

C. Transport layer to Network layer

D. Transport layer to Data link layer

E. Network layer to Data link layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer E: Network layer to Data link layer

ARP takes a logical IP address and resolves the physical MAC address, connecting network addressing to data-link addressing. Evidence: PDF 17, 15, 15.

Why each choice is right or wrong

A - Incorrect: Application to network omits the MAC destination of the mapping (PDF 17, 15, 15.)

B - Incorrect: Application to transport is not IP-to-MAC resolution (PDF 17, 15, 15.)

C - Incorrect: Transport ports are not the known IP address (PDF 17, 15, 15.)

D - Incorrect: Transport to data link incorrectly starts with ports (PDF 17, 15, 15.)

E - Correct: Network IP to data-link physical address matches both ends (PDF 17, 15, 15.)

Primary note proof

Study-note section NET20

> ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target's MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

> ARP - Address Resolution Protocol: It is used to
> resolve thephysical address of a device on a network,
> where its logical address is known.
> Physical address is the 48 bit address that is imprinted
> on theNIC or LAN card, Logical address is the Internet
> Address orcommonly known as IP address that is used
> to uniquely & universally identify a device.
> RARP- Reverse Address Resolution Protocol: It is
> used bya device on the network to find its Internet
> address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

> Network Layer
> • The network layer makes sure that the data is
> deliveredto the receiver despite multiple
> intermediate devices.

• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of

the data link layer is hop to hop transmission of frames.

Question sources

Mock exam 4 · Question 7

Question 22 | Q00206

What is the first usable host addressing of prefix 172.105.180.0/21?

A. 172.105.180.1

B. 172.105.170.1

C. 172.105.175.1

D. 172.105.176.1

E. 172.105.172.1

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer D: 172.105.176.1

A /21 mask is 255.255.248.0, giving third-octet blocks of eight. 180 lies in the 176-183 block; masking therefore yields network 172.105.176.0, not 172.105.180.0. The first usable ordinary host is one address above the network boundary: 172.105.176.1. NET24 explicitly explains this unaligned host/prefix example.

Why each choice is right or wrong

A - Incorrect: This address is inside the block but is not its first usable address; the block begins at third octet 176.

B - Incorrect: Third octet 170 lies below the 176-183 range of the actual /21 block.

C - Incorrect: Third octet 175 is outside the block, immediately below its network boundary.

D - Correct: The network is 172.105.176.0/21, so its first usable ordinary host is 172.105.176.1.

E - Incorrect: Third octet 172 belongs to a different /21 block and is outside the specified network.

Primary note proof

Study-note section NET24

Compute network=address AND mask. Host bits zero identify the block start; host bits one give broadcast. With 172.16.28.252/20, the third octet is grouped in blocks of 16, so the network is 172.16.16.0 and broadcast 172.16.31.255. A written host/prefix such as 172.105.180.0/21 need not be an aligned block label: masking yields 172.105.176.0/21.

Study-note section NET25

For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP

addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Question sources

Mock exam 4 · Question 8

Question 23 | Q00207

What is the special IP with all binary 1's used for?

A. A destination address to send to all devices on the same subnet

B. A destination address to send to all devices on another subnet

C. A destination address if a device is sending to itself

D. A source address when a device doesn't yet have an IP address

E. A source address if a device is sending to itself

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: Limited broadcast on the local subnet

All 32 one bits form 255.255.255.255, the limited-broadcast destination in NET23. Its scope is the local network; routers do not ordinarily forward it to another subnet. Self-reference uses loopback, while an unconfigured source can use 0.0.0.0. Thus the all-ones value is a destination broadcast, not either kind of source address.

Why each choice is right or wrong

A - Correct: The all-ones IPv4 address targets limited broadcast on the sender’s local network.

B - Incorrect: Limited broadcast is local in scope; it is not the destination for another subnet’s directed broadcast.

C - Incorrect: Sending to the host itself uses loopback, commonly 127.0.0.1, rather than the all-ones address.

D - Incorrect: An unconfigured source can use all-zero 0.0.0.0, not all-one 255.255.255.255.

E - Incorrect: The all-ones address is a broadcast destination, not a self-reference source address.

Primary note proof

Study-note section NET23

RFC1918 private blocks are 10.0.0.0/8, 172.16.0.0/12 (through 172.31.255.255), and 192.168.0.0/16. They can be reused internally but are not globally routed as public destinations. A LAN may also use legitimately assigned public addresses; private is common, not mandatory. 127.0.0.0/8 is IPv4 loopback, commonly 127.0.0.1, for local protocol access without an external link. 255.255.255.255 is limited broadcast; 0.0.0.0 can identify an unconfigured source. Link-local/APIPA uses 169.254.0.0/16, with edge /24 blocks excluded from normal automatic selection.

Study-note section NET11

Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP

addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Question sources

Mock exam 4 · Question 9

Question 24 | Q00208

Which of the following technologies can allow a host with IP address 10.0.0.1 to access the internet?

A. NAT

B. Route import

C. Dynamic route

D. Static route

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer A: NAT

The module prescribes translation of private internal addresses to a public address for Internet access. Conditional on the stem treating 10.0.0.1 as private, NAT is the offered mechanism. Evidence: PDF 19, 48, 19.

Why each choice is right or wrong

A - Correct: Private-to-public translation is exactly NAT (PDF 19, 48, 19.)

B - Incorrect: Importing routing information is not address translation (PDF 19, 48, 19.)

C - Incorrect: Dynamic path selection does not itself translate an internal address (PDF 19, 48, 19.)

D - Incorrect: Static routes do not themselves translate an internal address (PDF 19, 48, 19.)

Primary note proof

Study-note section NET29

A private host such as 10.0.0.1 commonly reaches the public Internet through NAT at a gateway, together with valid routing and a public external address. NAT alone does not supply a physical connection or DNS. A public address does not automatically grant permission to use it; both public and private unicast addresses can be used on a LAN under the correct allocation and subnet.

Study-note section NET30

Routing computes paths; forwarding chooses a next hop for an individual packet. Select the matching route with the longest prefix, then apply the routing system's tie-breaking criteria. A default /0 matches when nothing more specific does. Static routes are configured; dynamic protocols exchange reachability. Distance vector uses neighbor estimates, while link state distributes link information and computes shortest paths. Interior protocols operate within an autonomous system; BGP exchanges routes between systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of
network addresses in IPv4 led to the development of IPv6.
Due to this, we need to use private IP address inside the
organization and translate it to public IP address using
NAT.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local
network to a single IP address. This address is often used
by the router that connects the computers to the Internet.
The router can be connected to a DSL modem, cable
modem, T1 line, or even a dial-up modem. When other
computers on the Internet attempt to access computers
within the local network, they only see the IP address of
the router. This adds an extra level of security, since the
router can be configured as a firewall, only allowing
authorized systems to access the computers within the
network.

Data Communication and Computer Networking module | PDF page 19 | lines 52-64

ROUTING AND ROUTING PROTOCOLS
Routing is the act of moving information across an
internetwork from a source to a destination. At the heart of
such protocol is the routing algorithm that determines the
path for a packet.
• Routing requires a host or a router to have a
routingtable which is constructed by the routing
algorithm.

• There are wo kinds of routing algorithms (routing tables): nonadaptive (static) and adaptive (dynamic)
• And two kinds of protocols
o Interior routing protocols
o Exterior routing protocols

Question sources

Mock exam 4 · Question 10

Question 25 | Q00267

Which of the following protocol is used to define how messages are formatted and transmitted on the World Wide Web?

A. IP

B. FTP

C. SMTP

D. HTTP

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer D: HTTP

The question asks for the application protocol defining World Wide Web message exchange. HTTP defines web request/response semantics. IP supplies packet delivery beneath applications; FTP transfers files through its own protocol; SMTP submits and relays email. Those services do not define browser/server HTTP message formatting and exchange.

Why each choice is right or wrong

A - Incorrect: IP supplies best-effort packet delivery and does not define World Wide Web application-message semantics.

B - Incorrect: FTP is a file-transfer protocol with separate control and data connections, not the web-message protocol.

C - Incorrect: SMTP submits and relays mail rather than defining World Wide Web exchanges.

D - Correct: HTTP supplies web request/response semantics, matching the message formatting and transmission role asked about.

Primary note proof

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is

not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 23-31

Telnet 23 SNMP 161
SMTP 25 TFTP 69
HTTP 80 DNS 53
FTP 21 POP3 110
DNS 53 DHCP 68
HTTPS 443 NTP 123
SSH 22 RPC 530
Table. Sample TCP and UDP port numbers from well-known category

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Mock exam 4 · Question 69

Question 26 | Q00347

What is the bit pattern for the first octet of a class -B network address?

A. 110XXXXX

B. 1110XXXXX

C. 10XXXXXX

D. 0XXXXXXX

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer C: 10XXXXXX

In historical classful IPv4 addressing, Class B starts with the two fixed bits 10; the remaining six bits of the first octet vary. This gives first-octet decimal values 128-191. The alternatives instead begin with the Class C, Class D and Class A patterns respectively; B also prints too many total bits for an octet.

Why each choice is right or wrong

A - Incorrect: The fixed prefix 110 identifies historical Class C rather than Class B.

B - Incorrect: The fixed prefix 1110 identifies multicast Class D; the printed pattern also exceeds eight positions.

C - Correct: Class B fixes the first two bits to 10 and permits six remaining first-octet bits.

D - Incorrect: A leading zero identifies historical Class A, not Class B.

Primary note proof

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Study-note section NET22

> Write IPv4 as four decimal octets 0-255, or as 32 binary bits. Use weights 128,64,32,16,8,4,2,1 within each octet. Thus 11000001 10000011 00011011 11111111 is 193.131.27.255. A value such as 257 is impossible. Syntactic validity differs from assignability: an address ending .0 or .255 can be a host under some prefixes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

> Classful addressing : In classful addressing, the
> addressspace is divided into five classes: A, B, C,
> D, and E.
> • Netid and Hostid In classful addressing, an IP
> addressin class A, B, or C is divided into netid
> and hostid.
> Subnetting: It allows a network to be split into several
> parts for internal use but still act like a single network to
> the outsideworld.
> • To implement subnetting, the router needs a

subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science model exam · Question 78

University of Gondar model exam · Question 78

Question 27 | Q00348

Which OSI model layer is responsible for flow control?

A. Application Layer

B. Transport Layer

C. Session Layer

D. Presentation Layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Transport Layer

Among these four layers transport explicitly provides end-to-end flow control. Data link also provides flow control but is not offered. Evidence: PDF 16, 16, 16, 16.

Why each choice is right or wrong

A - Incorrect: Application provides access to resources (PDF 16, 16, 16, 16.)

B - Correct: Transport explicitly carries out flow control (PDF 16, 16, 16, 16.)

C - Incorrect: Session manages dialogs and checkpoints (PDF 16, 16, 16, 16.)

D - Incorrect: Presentation handles translation, compression and encryption (PDF 16, 16, 16, 16.)

Primary note proof

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET19

OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Data Communication and Computer Networking module | PDF page 16 | lines 18-32

Session Layer
• The session layer establishes a session between the communicating devices called dialog and synchronizes their interaction. It is the responsibility of the session layer to establish and synchronize the dialogs. It is also called the network dialog controller.
• The session layer at the sending side accepts data from the presentation layer adds checkpoints to it called sync bits and passes the data to the transport layer. At the receiving end the session layer receives

data from the transport layer removes the checkpoints inserted previously and passes the data to the presentation layer. So, the main responsibility of session layer is dialog control and synchronization.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Question sources

Computer Science model exam · Question 79

University of Gondar model exam · Question 79

Question 28 | Q00349

Which of the following networking device is used in physical layer?

A. Bridge

B. Router

C. Switch

D. Hub

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Hub

An active hub is a multiport repeater, and a repeater operates only at the physical layer. Evidence: PDF 12, 11, 12, 12, 12.

Why each choice is right or wrong

A - Incorrect: Bridge filters based on MAC (PDF 12, 11, 12, 12, 12.)

B - Incorrect: Router routes by IP (PDF 12, 11, 12, 12, 12.)

C - Incorrect: Switch filters/forwards by MAC (PDF 12, 11, 12, 12, 12.)

D - Correct: Active hub repeats physical signals (PDF 12, 11, 12, 12, 12.)

Primary note proof

Study-note section NET09

A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Study-note section NET10

A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET11

Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is
normallyused to create connections between stations in
a physical startopology.
HUB: Hubs are commonly used to connect segments
(devices)of a LAN. Hubs don‘t filter information; instead
broadcast incoming packets to all computers connected to
it. A hub contains multiple ports which support 8, 12 or
24 RJ-45 ports. Two types: Passive hub and Active hub. It
is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 11 | lines 54-55

Repeater: A repeater is a device that operates only in the physical
layer. A repeater receives a signal and, before it

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets
across anetwork.
Unlike a hub which simply replicates what it receives on

oneport onto all the other ports, a switch keeps a record of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the packetdirectly to the recipient device by looking up the MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device used tointerconnect different types of network that uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to forwardpackets.
• The list of routing tables provides directions to transfer the data to a particular network destination -various paths to forward the data.
• A router analyzes a destination IP address of a givenpacket and compares it with the routing table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 12 | lines 60-61

BRIDGE: Connects two LANS having the same protocol - (e.g. Ethernet or Token ring).

Question sources

Computer Science model exam · Question 80

University of Gondar model exam · Question 80

Question 29 | Q00350

Assuming that Company VWX has main site and remote site. The main site has three LANs with LAN 1 has 100 hosts, LAN 2 has 60 hosts, and LAN 3 has 29 hosts. The remote site has two LANs, each with 100 hosts. The network uses private addresses. The Internet service provider assigned the company the network 200.100.50.2/26.vThe remote site uses the network prefix 192.168.10.0/24. What subnets and masks can you use for the LANs at the remote site and conserve address space?

A. 192.168.10.0/25 and 192.168.10.128/25

B. 192.168.10.0/30 and 192.168.10.128/30

C. 192.168.10.64/26 and 192.168.10.192/26

D. 192.168.10.32/28 and 192.168.10.64/28

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer A: 192.168.10.0/25 and 192.168.10.128/25

Each remote LAN needs 100 hosts. /25 leaves seven bits (128 total addresses) and two nonoverlapping /25 blocks exactly partition the /24. A is the capacity-minimal offered division even without requiring a host-reservation formula. Evidence: PDF 19, 19.

Why each choice is right or wrong

A - Correct: 128 total per LAN can accommodate 100 and the blocks are disjoint (PDF 19, 19.)

B - Incorrect: Four total per /30 cannot hold 100 (PDF 19, 19.)

C - Incorrect: 64 total per /26 cannot hold 100 (PDF 19, 19.)

D - Incorrect: 16 total per /28 cannot hold 100 (PDF 19, 19.)

Primary note proof

Study-note section NET27

> For a /24 with LAN demands 100,60,29: allocate .0/25 (hosts .1-.126), .128/26 (.129-.190), and .192/27 (.193-.222); .224/27 remains. Two remote LANs of 100 each fit .0/25 and .128/25 within a separate /24. Count router interfaces as hosts if not already included in the demand.

Study-note section NET26

> For H hosts, choose the smallest h satisfying 2^h-2≥H, then p=32-h. For 24 hosts use /27; 52 or 57 use /26; 100 use /25; 500 use /23. For at least six equal subnets choose b=3. A Class C /24 split to /27 yields eight subnets. Always check both the number of subnets and the host capacity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address, the address space is 2N because each bit can have twodifferent values (0 or 1) and N bits can have 2N values.
• IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science model exam · Question 81

University of Gondar model exam · Question 81

Question 30 | Q00351

The main site uses the network prefix 192.168.15.0/24. What subnets and masks can you use to provide sufficient addresses for LANs at the main site and conserve address space?

A. 192.168.15.0/25 for LAN 1, 192.168.15.128/26 for LAN 2, and 192.168.15.192/27 for LAN

B. 192.168.15.0/26 for LAN 1, 192.168.15.128/26 for LAN 2, and 172.15.192.0/29 for LAN 3

C. 192.168.15.0/27 for LAN 1, 192.168.15.128/26 for LAN 2, and 172.15.192.0/25 for LAN 3

D. 192.168.15.0/100 for LAN 1, 192.168.15.128/60 for LAN 2, and 172.15.192.0/29 for LAN 3

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer A: 192.168.15.0/25 for LAN 1, 192.168.15.128/26 for LAN 2, and 192.168.15.192/27 for LAN

With Q29 context (100,60,29 hosts), A allocates 128,64,32 total addresses without overlap within 192.168.15.0/24. These are the smallest offered powers of two meeting the host counts; other choices fail total capacity, prefix syntax or membership. Evidence: PDF 19, 19.

Why each choice is right or wrong

A - Correct: The /25,/26,/27 division is sufficient and stays within the parent (PDF 19, 19.)

B - Incorrect: First /26 has only 64 total addresses for 100 hosts; third block is outside the parent (PDF 19, 19.)

C - Incorrect: First /27 has only 32 total for 100; third block is outside the parent (PDF 19, 19.)

D - Incorrect: /100 and /60 exceed the 32-bit IPv4 width; third block is outside the parent (PDF 19, 19.)

Primary note proof

Study-note section NET27

> VLSM assigns different masks to different LAN sizes. Sort demands largest first, round each to a power-of-two address block including reservations, allocate aligned nonoverlapping blocks within the parent, and leave the remaining space explicit.

Study-note section NET27

> For a /24 with LAN demands 100,60,29: allocate .0/25 (hosts .1-.126), .128/26 (.129-.190), and .192/27 (.193-.222); .224/27 remains. Two remote LANs of 100 each fit .0/25 and .128/25 within a separate /24. Count router interfaces as hosts if not already included in the demand.

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP
addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Question sources

Computer Science model exam · Question 82

University of Gondar model exam · Question 82

Question 31 | Q00352

Which network and mask would you use for the WAN link to save the most address space?

A. 192.168.11.240/27

B. 192.168.11.240/28

C. 192.168.11.240/29

D. 192.168.11.240/30

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: 192.168.11.240/30

Of the offered prefixes, /30 leaves the fewest host bits: two, giving four total addresses. It therefore consumes the least address space for a two-endpoint WAN under the conventional allocation implied by the options. Evidence: PDF 19, 19.

Why each choice is right or wrong

A - Incorrect: /27 consumes 32 addresses; .240 is also unaligned as a /27 network (PDF 19, 19.)

B - Incorrect: /28 consumes 16 addresses (PDF 19, 19.)

C - Incorrect: /29 consumes eight addresses (PDF 19, 19.)

D - Correct: /30 consumes four addresses and .240 is aligned (PDF 19, 19.)

Primary note proof

Study-note section NET28

> A conventional /30 has four addresses and two usable endpoints; check the link and addressing assumptions when selecting a WAN prefix. RFC3021 permits /31 on supported point-to-point links: both addresses are endpoints, so the subtract-two rule does not apply. /32 identifies one address/host route. Do not claim /30 is universally the smallest valid link prefix.

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet’s increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

> Address Space: An address space is the total
> number ofaddresses used by the protocol.
> • If a protocol uses N bits to define an address,
> the address space is 2N

because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP
addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Question sources

Computer Science model exam · Question 83

University of Gondar model exam · Question 89

Question 32 | Q00535

Among the following one helps to resolve internet name into IP Address.

A. Domain Name System

B. Address Resolution Protocol

C. User Datagram Protocol

D. Transmission Control Protocol

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer A: Domain Name System

DNS provides external naming in contrast to locally stored addresses. ARP instead resolves an already-known IP to a hardware address. Evidence: PDF 18, 65, 17, 18.

Why each choice is right or wrong

A - Correct: Domain Name Service is the naming mechanism (PDF 18, 65, 17, 18.)

B - Incorrect: ARP maps IP to physical address, not domain to IP (PDF 18, 65, 17, 18.)

C - Incorrect: UDP provides process delivery (PDF 18, 65, 17, 18.)

D - Incorrect: TCP provides reliable connection-oriented process delivery (PDF 18, 65, 17, 18.)

Primary note proof

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET20

> ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target's MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in

setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

/etc/hosts
The /etc/hosts file always contains the localhost IP
address, 127.0.0.1, which is
usedfor interprocess communication.
Never remove this line! Sometimes contains
addresses of additional hosts, whichcan be contacted
without using an external naming service such as
DNS (the Domain Name Server).

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to
resolve thephysical address of a device on a network,
where its logical address is known.
Physical address is the 48 bit address that is imprinted
on theNIC or LAN card, Logical address is the Internet
Address orcommonly known as IP address that is used
to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is
used bya device on the network to find its Internet
address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable
connection-oriented, reliable protocol. i.e. a connection is
established between the sender and receiver before the data can
be transmitted. It divides the data it receives from the upper
layer into segments and tags a sequence number to each
segment which is used at the receiving end for reordering of
data.
UDP - User Datagram Protocol: UDP is a simple protocol
used for process to process transmission. It is an unreliable,
connectionless protocol for applications that do not require flow

control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Question sources

Computer Science model exam · 2015 · Question 4

Question 33 | Q00536

You have been asked to create a subnet that supports 24 hosts. What subnet mask you use?

A. 255.255.255.224

B. 255.255.255.248

C. 255.255.255.240

D. 255.255.255.252

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer A: 255.255.255.224

A /27 mask (224=11100000) leaves five bits, giving 32 total addresses, enough for 24 hosts. All other options have fewer than 24 total addresses, so reservations do not affect the selection. Evidence: PDF 19, 19.

Why each choice is right or wrong

A - Correct: 224 leaves five bits and 32 addresses (PDF 19, 19.)

B - Incorrect: 248 leaves three bits and eight addresses (PDF 19, 19.)

C - Incorrect: 240 leaves four bits and 16 addresses (PDF 19, 19.)

D - Incorrect: 252 leaves two bits and four addresses (PDF 19, 19.)

Primary note proof

Study-note section NET26

For H hosts, choose the smallest h satisfying 2^h-2≥H, then p=32-h. For 24 hosts use /27; 52 or 57 use /26; 100 use /25; 500 use /23. For at least six equal subnets choose b=3. A Class C /24 split to /27 yields eight subnets. Always check both the number of subnets and the host capacity.

Study-note section NET24

A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP
addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Question sources

Computer Science model exam · 2015 · Question 5

Question 34 | Q00538

Assume you are asking to establish a network using 50(fifty) computers in mesh topology. How many cables required to establish a network?

A. 2225

B. 1225

C. 1450

D. 2450

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer B: 1225

Full-mesh cable count is n(n-1)/2, so 50×49/2=1225 distinct undirected cables. Each cable serves a pair and must not be counted twice. Evidence: PDF 23.

Why each choice is right or wrong

A - Incorrect: 2225 is not 50×49/2 (PDF 23.)

B - Correct: 1225 is the computed cable count (PDF 23.)

C - Incorrect: 1450 is not the pair count (PDF 23.)

D - Incorrect: 2450 counts both directions of each cable (PDF 23.)

Primary note proof

Study-note section NET08

> A full undirected mesh of n nodes needs n(n-1)/2 links and n-1 interfaces per node. For 50 nodes: 50×49/2=1225 links. Redundant paths improve fault tolerance at greater cable and interface cost.

Study-note section NET08

> Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 45-53

> Mesh Topology: in this topology devices are
> connected with many redundant interconnections
> between network nodes. In a full mesh topology,
> every node has a connection to every other node in
> the network, which makes it the most expensive
> topology over all the other topologies. The number of

cables grows fast as the number of nodes increases,
and it can be calculated by using the general formula
((n (n - 1)) /2), where n is the number of nodes in the

Question sources

Computer Science model exam · 2015 · Question 7

Question 35 | Q00552

A TCP/IP protocol for mapping an Internet Protocol address (IP address) to a physical machine address (MAC) that is recognized in the local network.

A. Address Resolution Protocol

B. Domain Name System

C. Revers Address Resolution Protocol

D. Internet Control Messaging Protocol

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A: Address Resolution Protocol

The known IP to MAC direction is explicitly ARP. Reverse ARP uses the opposite known and unknown addresses. Evidence: PDF 17, 18, 17.

Why each choice is right or wrong

A - Correct: ARP resolves physical from logical address (PDF 17, 18, 17.)

B - Incorrect: DNS is a name service, not IP-to-MAC resolution (PDF 17, 18, 17.)

C - Incorrect: RARP finds Internet address from physical address (PDF 17, 18, 17.)

D - Incorrect: ICMP signals transit problems (PDF 17, 18, 17.)

Primary note proof

Study-note section NET20

ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target’s MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Study-note section NET20

DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to
resolve thephysical address of a device on a network,
where its logical address is known.
Physical address is the 48 bit address that is imprinted
on theNIC or LAN card, Logical address is the Internet
Address orcommonly known as IP address that is used
to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is
used bya device on the network to find its Internet
address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Data Communication and Computer Networking module | PDF page 17 | lines 54-59

ICMP- Internet Control Message Protocol: It is a
signaling mechanism used to inform the sender about
datagram problemsthat occur during transit. It is used by
intermediate devices.
IGMP- Internet Group Message Protocol: It is a mechanism
that allows sending the same message to a group of recipients.

Question sources

Computer Science model exam · 2015 · Question 21

Question 36 | Q00553

One of the following is false the characteristics of controlled access in datalink layer.

A. Only one station is transmit at a time.

B. There will be a chance of collision.

C. Devices wishing to transmit must wait their turn.

D. Some deterministic networks use token passing.

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: There will be a chance of collision.

The module characterizes controlled access as permission-based and describes the medium-control objective as preventing simultaneous speakers. Thus the collision-chance characterization conflicts with its ideal controlled-access model. Evidence: PDF 21.

Why each choice is right or wrong

A - Incorrect: One transmitter at a time matches controlled permission (PDF 21.)

B - Correct: Colliding simultaneous access is inconsistent with the described controlled-access rule (PDF 21.)

C - Incorrect: Waiting for permission follows get permission (PDF 21.)

D - Incorrect: Token passing is explicitly in controlled access (PDF 21.)

Primary note proof

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Study-note section NET14

> Random access allows contention: ALOHA sends opportunistically; CSMA senses before sending. In legacy half-duplex Ethernet, CSMA/CD detects collision, sends a jam indication, stops, and retries after random backoff. The jam signals a collision, not its clearance. Wi-Fi 802.11 uses CSMA/CA with waiting/backoff and acknowledgments; sensing does not eliminate hidden-terminal collisions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 21 | lines 6-26

MEDIUM ACCESS CONTROL
The problem of controlling the access to the medium is similarto the rules of speaking in a meeting (the right to speak isupheld; two people do not speak at the same time; do notinterrupt each other; do not monopolize the discussion; ...) Three categories of multiple access protocols
Random Access Protocols - try your best like taxis do
• MA - Multiple Access
• CSMA - Carrier Sense MA
• CSMA/CD - CSMA with Collision Detection
• CSMA/CA - CSMA with Collision AvoidanceControlled-Access Protocols - get permission
• Reservation
• Polling
• Token Passing
Channelization Protocols - simultaneous use
• FDMA - Frequency-Division MA
• TDMA - Time-Division MA
• CDMA - Code-Division MA

Question sources

Computer Science model exam · 2015 · Question 22

Question 37 | Q00559

Among the following well known port number of TCP the port number of HTTP is:

A. 80

B. 25

C. 21

D. 110

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A: 80

The module port table pairs HTTP with TCP port 80. Evidence: PDF 28.

Why each choice is right or wrong

A - Correct: 80 is paired with HTTP (PDF 28.)

B - Incorrect: 25 is SMTP (PDF 28.)

C - Incorrect: 21 is FTP (PDF 28.)

D - Incorrect: 110 is POP3 (PDF 28.)

Primary note proof

Study-note section NET21

> Usual service ports: FTP control 21, SSH 22, Telnet 23, SMTP relay 25, DNS 53, HTTP 80, POP3 110, IMAP 143, HTTPS 443. Telnet provides unencrypted remote terminals; SSH protects remote access. Servers can be configured on other ports. A browser's HTTP exchange does not determine the physical duplex mode of its network interface.

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 23-31

> Telnet 23 SNMP 161
> SMTP 25 TFTP 69
> HTTP 80 DNS 53
> FTP 21 POP3 110
> DNS 53 DHCP 68
> HTTPS 443 NTP 123
> SSH 22 RPC 530
> Table. Sample TCP and UDP port numbers from well-known category

Question sources

Computer Science model exam · 2015 · Question 28

Question 38 | Q00560

Which organizational networking term describes exposing selected internal services to approved users outside the organization?

A. Internet

B. Extranet

C. intranet

D. All

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Extranet

Access scope determines the answer: an extranet extends selected organizational services to approved outsiders. An intranet serves members internally, while the Internet is the global network of networks. Neither organizational term determines geographic size or requires a particular physical connection. The question therefore asks about controlled external access rather than treating every external link as the same network category.

Why each choice is right or wrong

A - Incorrect: The Internet is the global network of networks; it does not specifically name controlled access to an organization's selected services.

B - Correct: An extranet exposes selected services to approved outsiders.

C - Incorrect: An intranet describes internal organizational access.

D - Incorrect: The three terms have different scopes, so all is not the requested category.

Primary note proof

Study-note section NET02

> An intranet serves an organization internally; an extranet exposes selected services to approved outsiders. Neither term alone specifies geographic size. A client requests a service; a server supplies it. Peer systems may perform both roles. Dedicated servers concentrate on service workloads but can still be clients of DNS, backup or update services. A ten-user cutoff is a teaching guideline, not a technical law. Centralization simplifies administration but creates dependencies; peer networks still need security.

Study-note section NET02

> PAN connects personal devices, such as Bluetooth accessories. LAN covers a local building or campus; MAN spans a metropolitan region; WAN connects distant regions. Device count alone does not determine the category. The Internet is a global network of networks, commonly the largest example of a WAN.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 2-19

> A network can be defined as two or more computers
> connected together in such a way that they can share
> resources. The primary purpose of a network is to share
> resources, and a resource could be:
> • a file,
> • a folder,
> • a printer,

• a disk drive, or just about anything else that exists on a computer.
Therefore, a computer network is simply a collection of computers or other hardware devices that are connected together, either physically or logically, using special hardware and software, to allow them to exchange information and cooperate. Networking is the term that describes the processes involved in designing, implementing, upgrading, managing and otherwise working with networks and network technologies.

Data Communication and Computer Networking module | PDF page 24 | lines 23-35

Network Architecture: based on the architecture (where do the clients get the shared resources?), networks can be categorized into three:
Client-Server Architecture: There can be one or more systems acting as Server. Other being Client, request the Server to serve requests. Servers take and process request on clients' behalf.
Peer-to-Peer (Point-to-point): Two systems can be connected Point-to-Point, or in other words back-to-back fashion. They both reside on same level and called peers.
There can be hybrid network which involves network architecture of both the above types.

Question sources

Computer Science model exam · 2015 · Question 29

Question 39 | Q00564

Class B private IP address space ranges:

A. 10.0.0.0----------19.255.255.255

B. 128.12.0.0--------191.24.255.255

C. 172.16.0.0--------172.31.255.255

D. 192.168.0.0------192.168.255.255

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer C: 172.16.0.0 through 172.31.255.255

The private allocation within the historical Class B range is 172.16.0.0/12. Its twelve fixed bits leave first octet 172 and second octets 16-31, inclusive. The 192.168 range is also private but lies in historical Class C space; the other printed ranges are not RFC1918 private allocations.

Why each choice is right or wrong

A - Incorrect: The private 10/8 block ends at 10.255.255.255; extending it through first octet 19 includes public space.

B - Incorrect: This broad portion of historical Class B space is not a designated RFC1918 private allocation.

C - Correct: The 172.16/12 private block spans exactly 172.16.0.0 through 172.31.255.255.

D - Incorrect: 192.168/16 is private, but it lies in historical Class C space rather than Class B.

Primary note proof

Study-note section NET23

> RFC1918 private blocks are 10.0.0.0/8, 172.16.0.0/12 (through 172.31.255.255), and 192.168.0.0/16. They can be reused internally but are not globally routed as public destinations. A LAN may also use legitimately assigned public addresses; private is common, not mandatory. 127.0.0.0/8 is IPv4 loopback, commonly 127.0.0.1, for local protocol access without an external link. 255.255.255.255 is limited broadcast; 0.0.0.0 can identify an unconfigured source. Link-local/APIPA uses 169.254.0.0/16, with edge /24 blocks excluded from normal automatic selection.

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

> Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
> • Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
> Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
> • To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex.

255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of network addresses in IPv4 led to the development of IPv6. Due to this, we need to use private IP address inside the organization and translate it to public IP address using NAT.

Question sources

Computer Science model exam · 2015 · Question 33

Question 40 | Q00570

Among the protocol one is used to provide remote access to servers and networking devices.

A. HTTP

B. SMTP

C. Telnet (terminal emulation protocol)

D. FTP

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer C: Telnet (terminal emulation protocol)

TELNET is explicitly the Virtual Terminal protocol, which fits terminal emulation for remote access to servers or devices. Evidence: PDF 18.

Why each choice is right or wrong

A - Incorrect: HTTP is not the listed virtual-terminal service (PDF 18.)

B - Incorrect: SMTP is electronic mail (PDF 18.)

C - Correct: TELNET is explicitly virtual terminal (PDF 18.)

D - Incorrect: FTP is file transfer (PDF 18.)

Primary note proof

Study-note section NET21

Usual service ports: FTP control 21, SSH 22, Telnet 23, SMTP relay 25, DNS 53, HTTP 80, POP3 110, IMAP 143, HTTPS 443. Telnet provides unencrypted remote terminals; SSH protects remote access. Servers can be configured on other ports. A browser's HTTP exchange does not determine the physical duplex mode of its network interface.

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Question sources

Computer Science model exam · 2015 · Question 39

Question 41 | Q00574

The function of protocols in transport layer (TCP and UDP) includes following except,

A. Segment upper-layer application data.

B. Establish end-to-end operations.

C. Provide flow control

D. Addressing packets

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Addressing packets

Transport segments upper data, provides end-to-end process operations, and may provide flow control through TCP. Logical packet addressing belongs to network delivery. Not every UDP operation provides flow control.

Why each choice is right or wrong

A - Incorrect: Segmentation is explicit transport work

B - Incorrect: Process-to-process delivery is end-to-end

C - Incorrect: Flow control is supported by the transport layer, though not UDP

D - Correct: IP packet addressing is assigned to the network layer

Primary note proof

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.

• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Question sources

Computer Science model exam · 2015 · Question 43

Question 42 | Q00579

For this hypothetical design, Debre Markos University’s main campus and its Burie agricultural branch are in different towns beyond a single metropolitan area. Which geographic network category connects the two campus LANs?

A. Local Area Network

B. Metropolitan Area Network

C. Personal Area Network

D. Wide Area Network

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: Wide Area Network

A WAN connects geographically distant networks across regions rather than covering one local site or one metropolitan area. Under the explicit hypothetical placement, the two campus LANs need a WAN connection. A LAN can serve each campus locally, while a MAN covers a metropolitan region and a PAN connects nearby personal devices. The conclusion follows from the stated geography, not the institutions' names or number of computers.

Why each choice is right or wrong

A - Incorrect: A LAN describes local campus connectivity, not the stated inter-town connection.

B - Incorrect: A MAN spans one metropolitan region, excluded by the stated geography.

C - Incorrect: A PAN connects personal nearby devices rather than distant campus networks.

D - Correct: A WAN connects the geographically separated campus LANs.

Primary note proof

Study-note section NET02

> PAN connects personal devices, such as Bluetooth accessories. LAN covers a local building or campus; MAN spans a metropolitan region; WAN connects distant regions. Device count alone does not determine the category. The Internet is a global network of networks, commonly the largest example of a WAN.

Study-note section NET02

> An intranet serves an organization internally; an extranet exposes selected services to approved outsiders. Neither term alone specifies geographic size. A client requests a service; a server supplies it. Peer systems may perform both roles. Dedicated servers concentrate on service workloads but can still be clients of DNS, backup or update services. A ten-user cutoff is a teaching guideline, not a technical law. Centralization simplifies administration but creates dependencies; peer networks still need security.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 26-44

> Personal Area Network (PAN): is a network may be
> spanned across a given table with distances between
> the devices not more than few meters. The technology
> used to interconnect the devices could be a Bluetooth.
> These networks are called Personal Area Networks,
> since the devices interconnected in these networks are
> belongs to a single person.
> Local Area Network (LAN): is a network that may

span across a building, or across several buildings within a single organization, by using intermediate devices, like switches and/or hubs, to interconnect devices in all floors.
Metropolitan Area Network (MAN): is a network that may span across a whole city interconnecting several buildings and organizations.
Wide Area Network (WAN): is a network that may span across multiple cities, or an entire country, or an entire continent or it may even cover the whole world. For example, an Internet is one example of WAN.

Question sources

Computer Science model exam · 2015 · Question 48

Question 43 | Q00582

In OSI reference model, which one is responsible for the translation of incoming and outgoing data .

A. Presentation layer.

B. Application layer.

C. Transport layer.

D. Network layer.

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: Presentation layer.

Representation translation is explicitly one of presentation-layer duties. Evidence: PDF 16, 16, 16, 15.

Why each choice is right or wrong

A - Correct: Presentation translates data representations (PDF 16, 16, 16, 15.)

B - Incorrect: Application provides network resource access (PDF 16, 16, 16, 15.)

C - Incorrect: Transport delivers between processes (PDF 16, 16, 16, 15.)

D - Incorrect: Network provides logical addressing and routes (PDF 16, 16, 16, 15.)

Primary note proof

Study-note section NET19

OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which

bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Question sources

Computer Science model exam · 2015 · Question 51

Question 44 | Q00583

The choice of one topology over the other has an impact on:

A. The capability of the network.

B. The growth of the network.

C. The type of equipment that the network needs.

D. All

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer D: All

Topology changes cable requirements, central equipment, failure behavior and installation/reconnection costs. The module examples support effects on capability, growth and required equipment. Evidence: PDF 23, 23, 23, 24.

Why each choice is right or wrong

A - Incorrect: Link and failure behavior affect capability (PDF 23, 23, 23, 24.)

B - Incorrect: Cable growth and reconnection costs affect expansion (PDF 23, 23, 23, 24.)

C - Incorrect: A star requires a hub/switch while a mesh requires many direct links (PDF 23, 23, 23, 24.)

D - Correct: All three effects follow from the contrasted layouts (PDF 23, 23, 23, 24.)

Primary note proof

Study-note section NET08

> Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Study-note section NET08

> A full undirected mesh of n nodes needs n(n-1)/2 links and n-1 interfaces per node. For 50 nodes: 50×49/2=1225 links. Redundant paths improve fault tolerance at greater cable and interface cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

> Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Data Communication and Computer Networking module | PDF page 23 | lines 4-18

Bus Topology: in this topology all devices are connected to a central cable, called the bus or backbone, which is terminated at its ends. The purpose of the terminators is to stop the signal from bouncing, thereby clearing the cable so that other computers can send data. Message transmitted along the Bus is visible to all computers connected to the backbone cable. As the message arrives at each workstation, the workstation checks the destination address contained in the message either to process or drop the packet if it matches or not respectively. Its advantages are, ease of installation and less amount of cable requirement. Its main drawback is, the entire network will be shut down if there is a break in the main cable.

Data Communication and Computer Networking module | PDF page 23 | lines 45-53

Mesh Topology: in this topology devices are connected with many redundant interconnections between network nodes. In a full mesh topology, every node has a connection to every other node in the network, which makes it the most expensive topology over all the other topologies. The number of cables grows fast as the number of nodes increases, and it can be calculated by using the general formula ((n (n - 1)) /2), where n is the number of nodes in the

Data Communication and Computer Networking module | PDF page 24 | lines 2-10

network. It has several benefits, such as: dedicated links between devices, robust (single link failure don‘t affect entire network), privacy/security (direct communication between communicating devices), and ease of fault identification and isolation. Its drawbacks include, installation and reconnection are difficult (large number of cables), huge amount of cables consumes a lot of space, and it is the most expensive of all.

Question sources

Computer Science model exam · 2015 · Question 52

Question 45 | Q00585

One of the following is half-duplex mode of transmission.

A. Walky-talky/policeman phone

B. Television

C. Telephone

D. One-Way Radio

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer A: Walky-talky/policeman phone

Half duplex permits both directions at different times. A conventional walkie-talkie alternates talk and listen, matching this definition. Evidence: PDF 5.

Why each choice is right or wrong

A - Correct: Alternating talk/listen matches half duplex (PDF 5.)

B - Incorrect: Ordinary television broadcast only receives, matching simplex (PDF 5.)

C - Incorrect: An ordinary two-way simultaneous telephone call is full duplex (PDF 5.)

D - Incorrect: One-way radio is simplex (PDF 5.)

Primary note proof

Study-note section NET06

> Simplex is one way, half duplex alternates directions, and full duplex permits simultaneous sending/receiving. A walkie-talkie is half duplex; a telephone conversation uses full duplex. Serial sends sequential bits; parallel uses multiple conductors and may suffer skew. Character-oriented asynchronous serial adds start/stop bits; synchronous framing uses ongoing clock recovery. Neither is synonymous with browser asynchronous requests.

Study-note section NET06

> Transmission delay is packet bits divided by link bit rate; propagation delay is distance divided by signal speed. Processing and queueing add delay. A 1000-byte packet takes 0.8 ms to serialize at 10 Mbit/s; increasing line rate does not reduce propagation time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 5 | lines 43-48

> • Simplex: Only one of the devices sends the data and the
> other one only receives the data (Unidirectional).
> • Half Duplex: both the stations can transmit as well as
> receive but not at the same time.

• Full duplex mode: both stations can transmit and receive at the same time.

Question sources

Computer Science model exam · 2015 · Question 54

Question 46 | Q00588

VPN is abbreviated as

A. Visual Private Network

B. Virtual Private Network

C. Virtual Protocol Network

D. Virtual Protocol Networking

Hint

Find the full expansion in the terminology note and compare each word rather than only matching initials.

Answer B: Virtual Private Network

NET11 expands VPN as Virtual Private Network and describes the logical private connection it provides over another network. The word Virtual matters because the connection is logically overlaid rather than necessarily a separate physical private network. The alternatives replace parts of the established expansion with visual or protocol and therefore do not match the note.

Why each choice is right or wrong

A - Incorrect: Visual substitutes an unrelated word for Virtual in the established VPN expansion.

B - Correct: Virtual Private Network is the exact expansion given in NET11.

C - Incorrect: Protocol replaces Private, changing the acronym expansion away from the defined term.

D - Incorrect: Protocol Networking is not the defined Private Network phrase in VPN.

Primary note proof

Study-note section NET11

A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Study-note section NET11

Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 2-19

A network can be defined as two or more computers connected together in such a way that they can share resources. The primary purpose of a network is to share resources, and a resource could be:
• a file,
• a folder,
• a printer,
• a disk drive, or just about anything else that exists on a computer.
Therefore, a computer network is simply a collection of computers or other hardware devices that are connected together, either physically or logically, using special hardware and software, to allow them to exchange information and cooperate. Networking is the term that describes the processes involved in designing, implementing, upgrading, managing and otherwise working with networks and network technologies.

Question sources

Computer Science model exam · 2015 · Question 57

Question 47 | Q00596

How many equal /27 subnets can be created from 192.168.20.0/24, counting every subnet?

A. 8

B. 16

C. 32

D. 64

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer A: 8

Subnet counts depend on both the parent and child prefixes. Dividing a /24 parent into /27 children borrows three bits, giving 2^3 = 8 subnets. Each child has five host bits and therefore 32 total addresses, but that address count is a different quantity. Counting every subnet avoids the obsolete practice of excluding the first and last subnet.

Why each choice is right or wrong

A - Correct: Three borrowed bits produce eight /27 subnets.

B - Incorrect: Sixteen would require four borrowed bits, not three.

C - Incorrect: Thirty-two is the number of addresses in each /27, not the number of children of this /24.

D - Incorrect: Sixty-four requires six borrowed bits rather than three.

Primary note proof

Study-note section NET26

> To split parent /a into equal /p blocks, borrow b=p-a bits and obtain 2^b subnets. Do not subtract two subnets under modern practice. For a Class B /16 parent split to /19, b=3 gives eight subnets, each with 8192 addresses and 8190 conventional hosts. A /19 alone does not state its parent.

Study-note section NET26

> For H hosts, choose the smallest h satisfying 2^h-2≥H, then p=32-h. For 24 hosts use /27; 52 or 57 use /26; 100 use /25; 500 use /23. For at least six equal subnets choose b=3. A Class C /24 split to /27 yields eight subnets. Always check both the number of subnets and the host capacity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

> IP addresses: IP address uniquely identifies a device on the
> Internet. Every host and router on the Internet has an IP address,
> which encodes its network number and host number.
> • The combination is unique: in principle, no two
> machines on the Internet have the same IP address.
> • An IPv4 address is 32 bits long
> • They are used in the Source address and Destination
> address fields of IP packets.
> • An IP address does not refer to a host but it refers to a
> network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

> Classful addressing : In classful addressing, the
> addressspace is divided into five classes: A, B, C,
> D, and E.
> • Netid and Hostid In classful addressing, an IP

addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Data Communication and Computer Networking module | PDF page 19 | lines 37-44

CIDR: A class B address is far too large for most
organizationsand a class C network, with 256 addresses is
too small. This leads to granting Class B address to
organizations who do not require all. This is resulting in
depletion of Address space. A solution is CIDR (Classless
InterDomain Routing) The basic idea behind CIDR, is to
allocate the remaining IP addresses in variable-sized
blocks, without regard to the classes.

Question sources

Computer Science model exam · 2015 · Question 65

Question 48 | Q00601

Given an IP Address 172.16.28.252 with the subnet mask of 255.255.240.0 what is the correct network address?

A. 172.16.24.0

B. 172.16.28.0

C. 172.16.0.0

D. 172.16.16.0

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer D: 172.16.16.0

240=11110000 in the third mask octet. 28 (00011100) AND 240 (11110000)=16 (00010000); the last masked octet is zero. Thus the network is 172.16.16.0. Evidence: PDF 18, 19, 19.

Why each choice is right or wrong

A - Incorrect: 24 is not aligned to a 16-wide third-octet block (PDF 18, 19, 19.)

B - Incorrect: 28 retains host bits instead of masking them (PDF 18, 19, 19.)

C - Incorrect: 0 would discard the retained third-octet network bit (PDF 18, 19, 19.)

D - Correct: 16 is the result of the mask operation (PDF 18, 19, 19.)

Primary note proof

Study-note section NET24

A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide's 255.248.0.0 mistake.

Study-note section NET24

Compute network=address AND mask. Host bits zero identify the block start; host bits one give broadcast. With 172.16.28.252/20, the third octet is grouped in blocks of 16, so the network is 172.16.16.0 and broadcast 172.16.31.255. A written host/prefix such as 172.105.180.0/21 need not be an aligned block label: masking yields 172.105.176.0/21.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011

11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science model exam · 2015 · Question 70

Question 49 | Q00602

One of the following is true about packet switching.

A. Packets travel around the network seeking out the most efficient route.

B. Packets should necessarily seek out the shortest route.

C. The message gets broken into small data packets.

D. All

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer C: Packet switching divides a message into packets

The defining packet-switching property is dividing data into packets for shared-link forwarding. Routing may seek good paths, but packets do not guarantee the most efficient or geometrically shortest path. C directly states the definition; D includes unsupported universal route claims and therefore fails.

Why each choice is right or wrong

A - Incorrect: Routing chooses next hops, but packet switching does not guarantee packets discover the globally most efficient route.

B - Incorrect: Packets are not guaranteed to take the geometrically shortest route.

C - Correct: Breaking a message into data packets is the packet-switching operation.

D - Incorrect: The all option includes unsupported route-optimality and necessary-shortest-path claims.

Primary note proof

Study-note section NET31

> Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Study-note section NET31

> Datagram packets are forwarded independently. Virtual circuits establish forwarding state, but resource reservation is optional, not a necessary property of every virtual circuit. Message switching stores a complete message before forwarding, without a dedicated end-to-end path. Packet store-and-forward waits for the full packet/frame; cut-through starts after enough header arrives; fragment-free historically waits through the initial collision-sensitive portion. No routing algorithm guarantees that every packet always takes the geometrically shortest route.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

> Network Layer
> • The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
> • The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
> • The network layer is responsible for

source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 20 | lines 27-36

Message Switching:
Packet Switching: Packet switching was designed to address the shortcomings of circuit switching in dealing with data communication. Unlike circuit switching where communication is continuous along a dedicated circuit, in packet switching, communication is discrete in form of packets.
The three switching modes to choose from are:
• Cut-through
• Store-and-forward
• Fragment-free

Question sources

Computer Science model exam · 2015 · Question 71

Question 50 | Q00603

Which statements correctly compare the seven-layer OSI reference model with the four-layer TCP/IP model? Select all that apply.

A. Both have application layers.

B. Both have comparable transport and network layers.

C. Both uses packet-switching instead of circuit-switching technology.

D. All

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A, B: Both have application layers.; Both have comparable transport and network layers.

Both models include application responsibilities and corresponding transport and network functions. TCP/IP groups the OSI session, presentation and application responsibilities into its application layer, and calls its network layer the Internet layer. An abstract reference model does not itself execute a switching technology. Consequently, the claim that both models necessarily use packet switching is not a valid shared model property.

Why each choice is right or wrong

A - Correct: Both models include an application layer, although TCP/IP folds several OSI upper-layer functions into it.

B - Correct: TCP/IP retains transport and maps OSI network responsibilities to its Internet layer.

C - Incorrect: A reference model does not itself use a switching technology or restrict every implementation to one.

D - Incorrect: The implemented-technology claim in C is not a property established for both abstract models.

Primary note proof

Study-note section NET19

> The four-layer TCP/IP model folds OSI layers 5-7 into application, retains transport, calls layer 3 Internet, and combines link/physical as network access. A model is a guide to functions, not a guarantee that every application gets a reliable network. Distinguish end-to-end application service from its underlying process-to-process transport.

Study-note section NET03

> A reference model divides responsibilities and interfaces; it is not itself executable networking software. OSI is a seven-layer reference model; TCP/IP describes an implemented protocol suite, usually taught with four layers. A five-layer teaching model separates physical and link functions. State the model before counting layers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 3-15

> The structure TCP/IP model is very similar to the
> structure of the OSI reference model. The OSI model has
> seven layers where the TCP/IP model has four layers.
> • The Application layer of TCP/IP model
> corresponds to the Application Layer of Session,
> Presentation & Application Layer of OSI model.
> • The Transport layer of TCP/IP model corresponds
> to theTransport Layer of OSI model
> • The Network layer of TCP/IP model corresponds
> to the Network Layer of OSI model

• The Host to network layer of TCP/IP model correspondsto the Physical and Data link Layer of OSI model.

Data Communication and Computer Networking module | PDF page 16 | lines 50-59

After understanding the concept of layered task and the OSI model, we introduce the TCP/IP model. This model is currently being used on our systems. TCP/IP model is a collection of protocols often called a protocol suite. It offers a rich variety of protocols.
It is a hierarchical model; it existed even before the OSI model was developed. Originally had four layers (bottom to top):
1. Host to Network Layer
2. Internet Layer
3. Transport Layer

Question sources

Computer Science model exam · 2015 · Question 72

Question 51 | Q00604

One of the following is network layer protocol.

A. HTTP

B. FTP

C. ARP

D. TCP

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: ARP

Within this module, ARP is grouped among protocols supporting IP at the network layer. This follows the assigned taxonomy rather than assuming that all taxonomies place ARP identically. Evidence: PDF 17, 17, 18, 18.

Why each choice is right or wrong

A - Incorrect: HTTP is an application service in the table (PDF 17, 17, 18, 18.)

B - Incorrect: FTP is application file transfer (PDF 17, 17, 18, 18.)

C - Correct: ARP is explicitly grouped under network/IP support (PDF 17, 17, 18, 18.)

D - Incorrect: TCP belongs to transport (PDF 17, 17, 18, 18.)

Primary note proof

Study-note section NET20

ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target's MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 22-38

Network Layer or IP
Also called as the Internetwork Layer (IP). It holds the IP protocol which is a network layer protocol and is responsiblefor source to destination transmission of data.
The Internetworking Protocol (IP) is a connection-less & unreliable protocol.
It is a best effort delivery service. i.e. there is no error checking in IP, it simply sends the data and relies on its underlying layers to get the data transmitted to the destination.Even if it is unreliable IP cannot be considered weak and useless; since it provides only the functionality that is requiredfor transmitting data thereby giving maximum efficiency.
Since there is no mechanism of error detection or correction in IP, there will be no delay introduced on a medium where thereis no error at all.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to resolve thephysical address of a device on a network, where its logical address is known.
Physical address is the 48 bit address that is imprinted on theNIC or LAN card, Logical address is the Internet Address orcommonly known as IP address that is used to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is used bya device on the network to find its Internet address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Question sources

Computer Science model exam · 2015 · Question 73

Question 52 | Q00629

One of the following is responsible for receiving and transmitting the signals across the physical media.

A. Datalink Layer

B. Physical layer

C. Application Layer.

D. Transport Layer.

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer B: Physical layer

Transmission of signals across a physical medium is explicitly the physical-layer task. Evidence: PDF 14, 15, 16, 16.

Why each choice is right or wrong

A - Incorrect: Data link frames data and controls access (PDF 14, 15, 16, 16.)

B - Correct: Physical converts bits to signals and sends them (PDF 14, 15, 16, 16.)

C - Incorrect: Application supplies user network services (PDF 14, 15, 16, 16.)

D - Incorrect: Transport manages process delivery (PDF 14, 15, 16, 16.)

Primary note proof

Study-note section NET09

> A NIC supplies a computer’s network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

> Physical Layer:
> • The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification

of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Question sources

Computer Science model exam · 2015 · Question 98

Question 53 | Q00636

Which listed device performs layer-1 signal regeneration without inspecting destination addresses?

A. Router

B. Switch

C. Firewall

D. Repeater

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Repeater

A repeater regenerates physical signals and therefore performs a layer-1 function. Routers decide where to forward network-layer packets, and ordinary Ethernet switches make link-layer decisions using MAC addresses. A firewall enforces a security policy rather than defining a physical signal-regeneration role. Replacing the duplicated firewall distractor restores a meaningful physical-layer choice without changing the layer-identification concept.

Why each choice is right or wrong

A - Incorrect: A router's defining forwarding function is network-layer IP routing rather than layer-1 signal repetition.

B - Incorrect: An ordinary Ethernet switch performs layer-2 MAC frame forwarding.

C - Incorrect: A firewall enforces traffic policy rather than being the layer-1 repeater/hub requested.

D - Correct: A repeater regenerates layer-1 signals without inspecting destination addresses.

Primary note proof

Study-note section NET09

> A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET11

> A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

> Active Hubs
> An active hub is actually a multipart repeater. It is
> normallyused to create connections between stations in

a physical startopology.
HUB: Hubs are commonly used to connect segments (devices)of a LAN. Hubs don‘t filter information; instead broadcast incoming packets to all computers connected to it. A hub contains multiple ports which support 8, 12 or 24 RJ-45 ports. Two types: Passive hub and Active hub. It is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets across anetwork.
Unlike a hub which simply replicates what it receives on oneport onto all the other ports, a switch keeps a record of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the packetdirectly to the recipient device by looking up the MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device used tointerconnect different types of network that uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to forwardpackets.
• The list of routing tables provides directions to transfer the data to a particular network destination -various paths to forward the data.
• A router analyzes a destination IP address of a givenpacket and compares it with the routing table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 11 | lines 54-55

Repeater: A repeater is a device that operates only in the physical layer. A repeater receives a signal and, before it

Question sources

Computer Science model exam · 2015 · Question 105

Question 54 | Q00637

A data communication component that helps to convert data into transmittable signal.

A. Transmission media

B. Transmitter

C. Receiver

D. Data source

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer B: Transmitter

NET01 separates creating data, converting it to a signal, carrying that signal and reconstructing the data. The stem asks specifically about the conversion into a transmittable signal, which is the transmitter's role. A source supplies the data before this conversion, while the receiver performs the reconstruction at the far end.

Why each choice is right or wrong

A - Incorrect: The transmission medium carries the signal; it is not the component that converts source data into it.

B - Correct: The transmitter converts data into a signal suitable for transmission over the medium.

C - Incorrect: The receiver reconstructs data from the incoming signal, performing the opposite end's role.

D - Incorrect: The data source creates the data; signal conversion is assigned separately to the transmitter.

Primary note proof

Study-note section NET01

> Data are raw representations; information is data interpreted for a purpose. Communication requires a message, sender, receiver, medium and agreed protocol. A source creates data; a transmitter converts it to a signal; a receiver reconstructs data. Distance affects engineering but is not a sixth component.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 10 | lines 36-55

> TRANSMISSION MEDIA
> Transmission media is a means by which a
> communicationsignal is carried from one system to
> another

A transmission medium can be defined as anything that cancarry information from a source to a destination. The transmission medium is usually free space, metallic cable orfiber optic cable. Guided and Unguided (WIRELESS) Guided Transmission Media uses a cabling system that guidesthe data signals along a specific path. Out of these twisted-paircable, coaxial cable transport signals in the form of electricsignals and fiber-optic cable transport signals in the form oflight. Unguided Transmission Media transport data without using a physical conductor. It uses wireless electromagnetic signals to send data. Three types of unguided media:
• Radio waves
• Micro waves
• Infrared.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 5 | lines 23-35

A Data Communication system has five components:-
• Message:- is the information to be communicated by the sender to the receiver.
• Sender The sender is any device that is capable of sending the data (message).
• Receiver: - is a device that the sender wants to communicate the data (message).
• Transmission Medium: - It is the path by which the

message travels from sender to receiver. It can be wired or wireless and many subtypes in both.
• Protocol:- It is an agreed upon set or rules used by the sender and receiver to communicate data. Rules that govern data communication.

Question sources

Computer Science model exam · 2015 · Question 106

Question 55 | Q00638

Among the following one is the responsibility of logical link control layer.

A. Identify the network layer protocol.

B. Marking the beginning and ending of the frame.

C. Addressing the frame.

D. All

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: Identify or multiplex the network-layer protocol

LLC identifies or multiplexes the upper-layer protocol carried over the link. NET13 assigns medium-dependent addressing/access and frame delivery to MAC, and says exact framing depends on the protocol. Therefore the specific LLC duty offered here is identifying the network-layer protocol; the other functions cannot all be attributed to LLC.

Why each choice is right or wrong

A - Correct: Identifying or multiplexing an upper-layer network protocol is an explicit LLC responsibility in NET13.

B - Incorrect: Beginning/end frame delimitation belongs to protocol-specific framing, not the distinguishing LLC upper-protocol identification function.

C - Incorrect: Medium-dependent frame addressing belongs to MAC rather than the LLC duty asked about.

D - Incorrect: The all option incorrectly includes frame addressing and delimitation as specifically LLC responsibilities.

Primary note proof

Study-note section NET13

Link-layer LLC identifies or multiplexes upper-layer protocols and may supply link control; MAC handles medium-dependent addressing/access and frame delivery. Network addressing belongs

to layer 3. Ethernet, PPP and HDLC are link protocols. Frame headers may carry addresses and control/type fields; synchronization and exact framing depend on the protocol.

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 21 | lines 6-26

MEDIUM ACCESS CONTROL
The problem of controlling the access to the medium is similarto the rules of speaking in a meeting (the right

to speak isupheld; two people do not speak at the same time; do notinterrupt each other; do not monopolize the discussion; ...) Three categories of multiple access protocols
Random Access Protocols - try your best like taxis do
• MA - Multiple Access
• CSMA - Carrier Sense MA
• CSMA/CD - CSMA with Collision Detection
• CSMA/CA - CSMA with Collision AvoidanceControlled-Access Protocols - get permission
• Reservation
• Polling
• Token Passing
Channelization Protocols - simultaneous use
• FDMA - Frequency-Division MA
• TDMA - Time-Division MA
• CDMA - Code-Division MA

Question sources

Computer Science model exam · 2015 · Question 107

Question 56 | Q00639

One of the following is ODD

A. 192.257.10.0

B. 192.168.10.0

C. 172.16.1.1

D. 192.168.0.0

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: 192.257.10.0

As a syntactic IPv4 address comparison, 257 exceeds the permitted 0-255 octet range. The others have four legal octets; no host mask is supplied to test host assignability. Evidence: PDF 19.

Why each choice is right or wrong

A - Correct: 257 violates the octet limit (PDF 19.)

B - Incorrect: All octets fall between 0 and 255 (PDF 19.)

C - Incorrect: All octets fall between 0 and 255 (PDF 19.)

D - Incorrect: All octets fall between 0 and 255 (PDF 19.)

Primary note proof

Study-note section NET22

> Write IPv4 as four decimal octets 0-255, or as 32 binary bits. Use weights 128,64,32,16,8,4,2,1 within each octet. Thus 11000001 10000011 00011011 11111111 is 193.131.27.255. A value such as 257 is impossible. Syntactic validity differs from assignability: an address ending .0 or .255 can be a host under some prefixes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

> Address Space: An address space is the total
> number ofaddresses used by the protocol.
> • If a protocol uses N bits to define an address,
> the address space is 2N
> because each bit can
> have twodifferent values (0 or 1) and N bits
> can have 2N values.
> • IPv4 uses 32-bit addresses, which means that
> the address space is 232 or 4,294,967,296
> (more than 4billion).
> Notations: There are two notations to show an IPv4 address:
> • Binary notation: The IPv4 address is displayed
> as 32bits. ex. 11000001 10000011 00011011
> 11111111
> • Dotted decimal notation: To make the IPv4
> address easier to read, Internet addresses are
> usually written in decimal form with a decimal
> point (dot) separatingthe bytes. Each byte (octet)
> is 8 bits hence each number in dotted-decimal
> notation is a value rangingfrom 0 to 255. Ex.
> 129.11.11.239

Question sources

Computer Science model exam · 2015 · Question 108

Question 57 | Q00640

The most commonly used method to convert analog data to digital form.

A. Line coding scheme

B. Data encoding

C. Pulse code modulation

D. Demodulation

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: Pulse code modulation (PCM)

NET15 identifies PCM as the analog-to-digital procedure of sampling the analog input, quantizing each sample and encoding its level as bits. The note does not establish comparative usage statistics, but PCM is the specific conversion method among the choices. Line coding maps existing digital bits to signals; demodulation recovers information from a carrier rather than necessarily digitizing analog data.

Why each choice is right or wrong

A - Incorrect: Line coding begins with digital bits and maps them to digital signal patterns, rather than digitizing analog input.

B - Incorrect: Data encoding is a broad description and does not identify the specific analog-digitization method supplied here.

C - Correct: PCM performs the sampling, quantization and binary encoding needed to turn analog input into digital data.

D - Incorrect: Demodulation removes carrier modulation; it does not itself specify analog-to-digital sampling and quantization.

Primary note proof

Study-note section NET15

> ASK varies carrier amplitude, FSK frequency, PSK phase; QAM combines amplitude and phase. AM/FM/PM can carry analog data. PCM digitizes analog input by sampling, quantizing and encoding. Sampling must exceed twice the highest retained baseband frequency in practical designs; b-bit samples have 2^b quantization levels and uncompressed rate fs×b. PAM/PWM are pulse modulation methods; pulse amplitude or width alone is not a binary PCM code.

Study-note section NET15

> Line coding maps digital bits to digital signal patterns. NRZ uses levels without a forced mid-bit return; Manchester supplies a mid-bit transition for clocking, with bit convention specified. Differential Manchester encodes using transitions relative to previous state. AMI alternates the polarity of successive ones; pseudoternary alternates zero pulses. Block coding adds code redundancy; scrambling replaces problematic patterns without necessarily increasing bit count. Multilevel schemes include 2B1Q and 8B6T; MLT-3 uses a sequence of voltage transitions.

Study-note section NET11

> A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and

demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 13 | lines 78-83

Analog to Digital Conversion using modulation
The definition of the term modulation is described in the next section. Here we have modulation techniques:
• Pulse Amplitude Modulation (PAM)
• Pulse Code Modulation (PCM)
• Pulse Width Modulation (PWM)

Data Communication and Computer Networking module | PDF page 13 | lines 42-50

Digital Data to Digital Signal conversion
Coding methods are used to convert digital data into digital signals. There are two types of coding methods:
• Line Coding
• Block Coding
Scrambling is also one of the ways to convert digital data to digital signals but is not used.
Line Coding: It is the process of converting Digital data into digital signal.

Data Communication and Computer Networking module | PDF page 13 | lines 65-77

• Digital modulation is used to convert digital data to analog signal. Ex ASK, FSK, PSK.
Analog Modulation can be accomplished in three ways:
• Amplitude modulation (AM)
• Frequency modulation (FM)
• Phase modulation (PM).
Digital Modulation Types (Digital to Analog signal conversion)
Digital modulation is used to convert digital data to analog signal. It can be accomplished in the following ways:
• Amplitude Shift Keying(ASK)
• Frequency Shift keying (FSK)
• Phase Shift keying (PSK)

Question sources

Computer Science model exam · 2015 · Question 109

Question 58 | Q00647

In a peer-to-peer architecture, which statements are false? Select all that apply. Interpret C as absence of a required client-server hierarchy.

A. Computers can function as both a client and a server.

B. A dedicated central administrator is required by peer-to-peer architecture.

C. There is no a hierarchy among the computers.

D. In peer - to - peer network security is not an issue.

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B, D: A dedicated central administrator is required by peer-to-peer architecture.; In peer - to - peer network security is not an issue.

Peers may request and supply services without a required client-server hierarchy. That capability does not impose a dedicated central administrator, although people may still take responsibility for administration. Peer systems also continue to need security because they expose data and services. The false statements therefore concern mandatory centralized administration and the dismissal of security, while the capability and no-required-hierarchy statements are true.

Why each choice is right or wrong

A - Incorrect: True: peers can both request and supply services, so this is not a false statement.

B - Correct: False: a dedicated central administrator is not a requirement of peer architecture.

C - Incorrect: True: peers do not require a client-server hierarchy, so this is not a false statement.

D - Correct: False: peer networks still require security.

Primary note proof

Study-note section NET02

> An intranet serves an organization internally; an extranet exposes selected services to approved outsiders. Neither term alone specifies geographic size. A client requests a service; a server supplies it. Peer systems may perform both roles. Dedicated servers concentrate on service workloads but can still be clients of DNS, backup or update services. A ten-user cutoff is a teaching guideline, not a technical law. Centralization simplifies administration but creates dependencies; peer networks still need security.

Study-note section NET33

> First identify the exact layer, protocol and scope. Flow/error control can exist at both link and transport layers; MAC addresses can be locally administered; bridges and switches both operate at layer 2. A LAN can use public or private unicast space. UDP and IP are both connectionless. Session dialogs and TCP connections are distinct senses of connection. Distinguish the protocol, layer and scope before treating these descriptions as exclusive.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 24 | lines 23-35

> Network Architecture: based on the architecture (where do the clients get the shared resources?), networks can be categorized into three:
> Client-Server Architecture: There can be one or more systems acting as Server. Other being Client, request the Server to serve requests. Servers take and process request on clients‘ behalf.
> Peer-to-Peer (Point-to-point): Two systems can be connected Point-to-Point, or in other words back-to-back fashion. They both reside on same level and called peers.
> There can be hybrid network which involves network architecture of both the above types.

Data Communication and Computer Networking module | PDF page 32 | lines 24-41

> A Windows workgroup is a group of standalone computers in a peer-to-peer network. Each computer in the workgroup uses its own local accounts database to authenticate resource access. The computers in a workgroup also do not have a common authentication process. The default-networking environment for a clean windows load is workgroup
> In general, a given Windows workgroup environment can contain many computers but work best with 15 or fewer computers. As the number of computers increases, a workgroup eventually become very difficult to administer and should be re-organized into multiple networks or set up as a client-server network.
> The computers in a workgroup are considered peers because they are all equal and share resources among each other without requiring a server. Since the workgroup doesn‘t share a common security and resource database, users and resources must be defined on each computer.

Question sources

Computer Science model exam · 2015 · Question 116

Question 59 | Q00653

Which of the following protocols uses both TCP and UDP?

A. FTP

B. SMTP

C. Telnet

D. DNS

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer D: DNS

Inspection of the module port table places DNS 53 in both TCP and UDP columns. The other named services occur only in its TCP column. Evidence: PDF 28, 18.

Why each choice is right or wrong

A - Incorrect: FTP 21 is listed in TCP only (PDF 28, 18.)

B - Incorrect: SMTP 25 is listed in TCP only (PDF 28, 18.)

C - Incorrect: Telnet 23 is listed in TCP only (PDF 28, 18.)

D - Correct: DNS 53 appears in both table columns (PDF 28, 18.)

Primary note proof

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET-S02

> In the conventional service bindings, FTP control, SMTP mail transfer and Telnet remote terminals use TCP. DNS supports both UDP and TCP on port 53; it is not exclusively a UDP service. The application protocol’s service role is distinct from the transport carrying its messages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 23-31

> Telnet 23 SNMP 161
> SMTP 25 TFTP 69
> HTTP 80 DNS 53
> FTP 21 POP3 110
> DNS 53 DHCP 68

HTTPS 443 NTP 123
SSH 22 RPC 530
Table. Sample TCP and UDP port numbers from well-known category

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Computer Science model exam · 2015 · Question 122

Question 60 | Q00654

Which single option gives the complete set of listed devices whose ordinary forwarding function is at the data-link layer?

A. Switch

B. Access point

C. Hub

D. A & B

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: A & B

An ordinary Ethernet switch forwards frames at layer 2, and an ordinary wireless access point bridges wireless and wired link-layer traffic. A hub simply repeats physical signals at layer 1. The question requests the complete set, so either switch or access point alone omits another qualifying device. The combined response is therefore the one complete selection.

Why each choice is right or wrong

A - Incorrect: A switch is layer 2, but this selection omits the qualifying access point.

B - Incorrect: A bridging access point is layer 2, but this selection omits the switch.

C - Incorrect: A hub repeats physical signals and does not inspect destination MAC addresses as a data-link device does.

D - Correct: Both the ordinary switch and bridging access point are data-link devices; this gives the complete set.

Primary note proof

Study-note section NET10

A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET09

A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets
across anetwork.
Unlike a hub which simply replicates what it receives on
oneport onto all the other ports, a switch keeps a record
of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the
packetdirectly to the recipient device by looking up the
MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is
normallyused to create connections between stations in
a physical startopology.
HUB: Hubs are commonly used to connect segments
(devices)of a LAN. Hubs don‘t filter information; instead
broadcast incoming packets to all computers connected to
it. A hub contains multiple ports which support 8, 12 or
24 RJ-45 ports. Two types: Passive hub and Active hub. It
is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical

layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Question sources

Computer Science model exam · 2015 · Question 123

Question 61 | Q00656

Which statement correctly describes the OSI reference model?

A. It has fewer layer that TCP/IP model.

B. The model itself is executable software installed to implement a network.

C. Used as a guide to help students or learners understand the communication process.

D. All

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: Used as a guide to help students or learners understand the communication process.

OSI divides communication responsibilities into seven layers and serves as a framework for explaining interfaces and communication processes. The usual TCP/IP teaching model has four layers, so OSI does not have fewer. A reference model can guide design, but it is not itself executable networking software or an installed protocol implementation. That distinction makes the learning-guide statement the correct description.

Why each choice is right or wrong

A - Incorrect: OSI has seven layers, more than the usual four-layer TCP/IP teaching model.

B - Incorrect: A reference model is not itself executable networking software, although it can guide design.

C - Correct: The model helps learners understand layered communication responsibilities.

D - Incorrect: The incorrect layer count and executable-software claim prevent all statements from being true.

Primary note proof

Study-note section NET03

> A reference model divides responsibilities and interfaces; it is not itself executable networking software. OSI is a seven-layer reference model; TCP/IP describes an implemented protocol suite, usually taught with four layers. A five-layer teaching model separates physical and link functions. State the model before counting layers.

Study-note section NET19

> The four-layer TCP/IP model folds OSI layers 5-7 into application, retains transport, calls layer 3 Internet, and combines link/physical as network access. A model is a guide to functions, not a guarantee that every application gets a reliable network. Distinguish end-to-end application service from its underlying process-to-process transport.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 12-29

> The Open Systems Interconnection (OSI) Model was
> developed by International Organization for Standardization
> (ISO). ISO is the organization, OSI is the model
> It was developed to allow systems with different platforms
> (Hardware & software) to communicate with each other.
> Now, it is a network model that defines the protocols for
> network communications.
> It is a hierarchical model that groups its processes into seven
> (7) layers. (Top to Bottom)
> 1. Application Layer
> 2. Presentation Layer

3. Session Layer
4. Transport Layer
5. Network Layer
6. Data Link Layer
7. Physical Layer
Each layer has specific functions to perform and has to co-operate with the layers above and below it.

Data Communication and Computer Networking module | PDF page 17 | lines 3-15

The structure TCP/IP model is very similar to the structure of the OSI reference model. The OSI model has seven layers where the TCP/IP model has four layers.
• The Application layer of TCP/IP model corresponds to the Application Layer of Session, Presentation & Application Layer of OSI model.
• The Transport layer of TCP/IP model corresponds to theTransport Layer of OSI model
• The Network layer of TCP/IP model corresponds to the Network Layer of OSI model
• The Host to network layer of TCP/IP model correspondsto the Physical and Data link Layer of OSI model.

Data Communication and Computer Networking module | PDF page 25 | lines 17-27

TCP/IP is a hierarchical protocol made up of interactive modules, each of which provides a specific functionality; however, the modules are not necessarily interdependent. Whereas the OSI model specifies which functions belong to each of its layers, the layers of TCP/IP suite contain relatively independent protocols that can be mixed and matched depending on the needs of the system. The term hierarchical means that each upper-level protocol is supported by one or more lower-level protocols.

Question sources

Computer Science model exam · 2015 · Question 125

Question 62 | Q00699

Which listed item is not one of the five basic components of data communication?

A. Message

B. Medium

C. Sender

D. Distance

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: Distance

The five basic components are message, sender, receiver, medium and protocol. Message, medium and sender therefore belong to the required set. Distance affects transmission engineering, such as propagation delay and attenuation, but it is not a sixth component in that model. Replacing the original protocol option with distance supplies a meaningful exception while retaining the negative component-identification task.

Why each choice is right or wrong

A - Incorrect: Message is explicitly one of the required components.

B - Incorrect: The medium carries signals and is explicitly a communication component.

C - Incorrect: The sender is explicitly included among communication components.

D - Correct: Distance affects engineering but is not one of the five basic communication components.

Primary note proof

Study-note section NET01

> Data are raw representations; information is data interpreted for a purpose. Communication requires a message, sender, receiver, medium and agreed protocol. A source creates data; a transmitter converts it to a signal; a receiver reconstructs data. Distance affects engineering but is not a sixth component.

Study-note section NET06

> Transmission delay is packet bits divided by link bit rate; propagation delay is distance divided by signal speed. Processing and queueing add delay. A 1000-byte packet takes 0.8 ms to serialize at 10 Mbit/s; increasing line rate does not reduce propagation time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 5 | lines 23-35

A Data Communication system has five components:-
• Message:- is the information to be communicated by the sender to the receiver.
• Sender The sender is any device that is capable of sending the data (message).
• Receiver: - is a device that the sender wants to communicate the data (message).
• Transmission Medium: - It is the path by which the message travels from sender to receiver. It can be wired or wireless and many subtypes in both.
• Protocol:- It is an agreed upon set or rules used by the sender and receiver to communicate data. Rules that govern data communication.

Question sources

Computer Science model exam 2 · Question 40

Question 63 | Q00700

There are two types of network topologies from these topologies which one of the following defines how hosts access the media to send data.

A. Logical Topology

B. Bus Topology

C. Physical Topology

D. Star Topology

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer A: Logical topology

The distinction is between physical layout and communication/access relationships. NET08 defines logical topology in the latter sense, matching how hosts obtain medium access. Bus and star are particular arrangements rather than the general physical-versus-logical category requested.

Why each choice is right or wrong

A - Correct: Logical topology describes communication and medium-access relationships among hosts.

B - Incorrect: Bus is a particular shared-backbone arrangement, not the general category defining access relationships.

C - Incorrect: Physical topology describes link layout and connections rather than the logical rules for medium access.

D - Incorrect: Star describes endpoints around a central device; it does not name the general access-topology category.

Primary note proof

Study-note section NET08

Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Study-note section NET08

Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 11 | lines 34-51

NETWORK TOPOLOGY
A network topology is a description of the layout of the region or area covered by that network.
There are two types of connections. They are: point-to-point and multi-point.
Point-to-point connections provide a direct link between two devices.
Multi-point connections provide a link between three or more devices on a network.
There are seven basic topologies in the study of network topology:
• Point-to-point topology,
• Bus (point-to-multipoint) topology,
• Ring topology,
• Star topology,
• Hybrid topology,
• Mesh topology and,
• Tree topology.

Data Communication and Computer Networking module | PDF page 22 | lines 45-52

Inter-connectivity: components of a network, including end devices and interconnecting devices, can be connected to each other differently in some fashion. By connectedness we mean either logically, physically or both ways. Network topology refers to the shape of a network, or the network‘s layout. It is

the geometric representation of the relationship of all
the links and linking devices to one another. There are

Question sources

Computer Science model exam 2 · Question 41

Question 64 | Q00701

Segmentation of a data stream happens at which layer of the OSI model?

A. Physical

B. Data Link

C. Network

D. Transport

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Transport

Transport explicitly divides data into segments and reassembles sequence-numbered segments. Evidence: PDF 16, 14, 15, 15.

Why each choice is right or wrong

A - Incorrect: Physical delivers bits (PDF 16, 14, 15, 15.)

B - Incorrect: Data link handles frames (PDF 16, 14, 15, 15.)

C - Incorrect: Network handles packets (PDF 16, 14, 15, 15.)

D - Correct: Transport explicitly performs segmentation (PDF 16, 14, 15, 15.)

Primary note proof

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in

setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical

layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow

controlor error control
• The main responsibility of Network Layer is
transmission of packets from source to destination

Question sources

Computer Science model exam 2 · Question 42

Question 65 | Q00702

When data is encapsulated, which is the correct order?

A. Data, frame, packet, segment, bit

B. Data, segment, packet, frame, bit

C. Segment, data, packet, frame, bit

D. Data, segment, frame, packet, bit

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Data, segment, packet, frame, bit

Follow the sending path downward: upper data becomes transport segments, network packets, data-link frames, then physical bits. Evidence: PDF 16, 15, 15, 14.

Why each choice is right or wrong

A - Incorrect: Frames cannot precede network packets on the sending path

B - Correct: The encapsulation order is data, segment, packet, frame, then bit

C - Incorrect: Segmentation follows receipt of upper-layer data

D - Incorrect: Network packet must precede data-link frame

Primary note proof

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits

into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Question sources

Computer Science model exam 2 · Question 43

Question 66 | Q00703

You want to implement a network medium that is not susceptible to EMI. Which type of cabling should you use?

A. Coaxial

B. Category 6 UTP

C. Microwave

D. Fiber-optic

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer D: Fiber-optic cable

NET07 identifies fiber as a light-guiding medium that resists electromagnetic interference. Coax and UTP carry electrical signals and can be affected by coupled interference. Microwave is wireless, so it is not a cabling answer even before considering interference in radio transmission.

Why each choice is right or wrong

A - Incorrect: Coax has shielding but still carries electrical signals; it does not offer fiber's immunity to electromagnetic coupling.

B - Incorrect: Twisted pairs reduce interference but unshielded copper cable remains susceptible to electromagnetic coupling.

C - Incorrect: Microwave is an unguided wireless medium, not a cable matching the requested cabling choice.

D - Correct: Fiber carries light and resists electromagnetic interference rather than conducting the unwanted electrical coupling.

Primary note proof

Study-note section NET07

> Twisted pairs reduce coupled interference; UTP has no added shield and STP adds shielding. Coax has an inner conductor and outer shield sharing an axis. Fiber guides light and resists electromagnetic interference; it supports high capacity and long reach, but requires suitable optics and installation. Exact speed, cost and reach depend on the specification, not the material alone. Fiber can still be tapped.

Study-note section NET07

> Radio, microwave and infrared are unguided media. Ground waves follow Earth, sky waves can reflect from ionized layers, and line-of-sight paths require suitable visibility. Microwaves are radio waves in the broader physical classification. Infrared generally does not penetrate walls. Directionality depends on antennas: omnidirectional means broad azimuth coverage, not equal radiation in every 3-D direction. Legacy Ethernet straight-through cables join unlike MDI/MDI-X

roles; crossover joins like roles, such as PC-PC or switch-switch. Auto-MDI-X can remove that requirement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 10 | lines 36-55

TRANSMISSION MEDIA
Transmission media is a means by which a communicationsignal is carried from one system to another
A transmission medium can be defined as anything that cancarry information from a source to a destination. The transmission medium is usually free space, metallic cable orfiber optic cable. Guided and Unguided (WIRELESS) Guided Transmission Media uses a cabling system that guidesthe data signals along a specific path. Out of these twisted-paircable, coaxial cable transport signals in the form of electricsignals and fiber-optic cable transport signals in the form oflight.
Unguided Transmission Media transport data without using a physical conductor. It uses wireless electromagnetic signals to send data. Three types of unguided media:
• Radio waves
• Micro waves
• Infrared.

Data Communication and Computer Networking module | PDF page 11 | lines 23-33

Twisted-pair cable: Consists of two conductors (normally copper), each with its own plastic insulation, twisted together.
Two types
• Unshielded Twisted-Pair cable (UTP)
• Shielded Twisted-Pair cable (STP)
Coaxial cable:- uses copper wire to conduct the signals electronically
Fiber-optic cable:- uses glass or plastic conductor and transmits the signals as light
Wireless/Unguided media the signal propagates in the atmosphere and in outer space

Question sources

Computer Science model exam 2 · Question 44

Question 67 | Q00704

Which responsibility belongs to the OSI network layer rather than the data-link layer?

A. logical link control

B. Network Addressing

C. media access control

D. Hardware Addressing

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Network Addressing

Logical network addressing, such as IP addressing, supports delivery across interconnected networks and belongs to the network layer. The data-link layer instead handles local frame delivery, hardware/MAC addressing, medium access and link-control functions. LLC identifies or multiplexes upper-layer protocols and may supply link control. The term hardware addressing here denotes frame-level MAC addressing, not an IP address.

Why each choice is right or wrong

A - Incorrect: LLC is a data-link sublayer role for upper-protocol identification and link control.

B - Correct: Logical network addressing is a network-layer responsibility.

C - Incorrect: Medium access control is a data-link role.

D - Incorrect: MAC hardware addressing is used for link-frame delivery.

Primary note proof

Study-note section NET13

> Link-layer LLC identifies or multiplexes upper-layer protocols and may supply link control; MAC handles medium-dependent addressing/access and frame delivery. Network addressing belongs to layer 3. Ethernet, PPP and HDLC are link protocols. Frame headers may carry addresses and control/type fields; synchronization and exact framing depend on the protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

> Data Link Layer
> • The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
> • Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames

and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Question sources

Computer Science model exam 2 · Question 45

Question 68 | Q00705

Which class of IP address provides a maximum of only 254 host addresses per network ID?

A. Class A

B. Class D

C. Class B

D. Class C

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer D: Class C under the historical default mask

The wording asks about historical address classes. Class C uses default /24, leaving eight host bits: 2^8−2=254 ordinary usable hosts. Classes A and B leave more bits, while Class D identifies multicast rather than an ordinary unicast host network. Under CIDR the actual prefix would replace these defaults.

Why each choice is right or wrong

A - Incorrect: Default Class A /8 leaves 24 host bits, far more than 254 usable hosts.

B - Incorrect: Class D is multicast space, not an ordinary classful unicast network with 254 hosts.

C - Incorrect: Default Class B /16 leaves sixteen host bits and 65534 ordinary usable hosts.

D - Correct: Default Class C /24 leaves eight host bits, giving 256 addresses minus network and broadcast.

Primary note proof

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet’s increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 37-44

CIDR: A class B address is far too large for most organizationsand a class C network, with 256 addresses is too small. This leads to granting Class B address to organizations who do not require all. This is resulting in depletion of Address space. A solution is CIDR (Classless InterDomain Routing) The basic idea behind CIDR, is to allocate the remaining IP addresses in variable-sized blocks, without regard to the classes.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address, the address space is 2N because each bit can have twodifferent values (0 or 1) and N bits can have 2N values.
• IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a

subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science model exam 2 · Question 46

Question 69 | Q00706

Which item is not a parameter or spatial characteristic of a continuous sinusoidal carrier wave?

A. Amplitude

B. Wavelength

C. phase

D. Discrete pulse

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer D: Discrete pulse

A sine wave has amplitude, frequency and phase. Given propagation speed, wavelength follows from frequency and describes its spatial repetition. A discrete pulse is a different waveform description, not a parameter of a continuous sine wave. This distinction concerns the specified carrier model; it does not claim that all pulse modulation is digital, since pulse amplitude and width can vary to carry analog information.

Why each choice is right or wrong

A - Incorrect: Amplitude specifies the sinusoid’s magnitude.

B - Incorrect: Wavelength specifies the spatial repetition of the propagating sinusoid.

C - Incorrect: Phase specifies displacement within the sinusoid’s cycle.

D - Correct: A discrete pulse is a waveform description rather than a sinusoidal parameter.

Primary note proof

Study-note section NET04

Analog signals vary continuously; digital signals use discrete symbol states. Data and signal types are separate: digital bits can modulate an analog carrier. A sine wave has amplitude, frequency and phase. Frequency f counts cycles/second, period T=1/f, and wavelength λ=v/f depends on propagation speed v. Phase specifies displacement within a cycle. Composite signals combine frequencies.

Study-note section NET15

Line coding maps digital bits to digital signal patterns. NRZ uses levels without a forced mid-bit return; Manchester supplies a mid-bit transition for clocking, with bit convention specified. Differential Manchester encodes using transitions relative to previous state. AMI alternates the polarity of successive ones; pseudoternary alternates zero pulses. Block coding adds code redundancy; scrambling replaces problematic patterns without necessarily increasing bit count. Multilevel schemes include 2B1Q and 8B6T; MLT-3 uses a sequence of voltage transitions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 7 | lines 40-48

ANALOG SIGNAL
An analog signal has infinitely many levels of intensity over a period of time.
A simple analog signal is a sine wave that cannot be further decomposed into simpler signals.
A sine wave is characterized by three parameters:
• Peak Amplitude
• Frequency
• Phase

Data Communication and Computer Networking module | PDF page 7 | lines 73-82

Wavelength
The wavelength of a signal refers to the relationship between frequency (or period) and propagation speed of the wave through a medium.
The wavelength is the distance a signal travels in one period.
It is given by
Wavelength = Propagation Speed X Period
OR
Wavelength =Propagation Speed X 1/Frequency

Data Communication and Computer Networking module | PDF page 7 | lines 89-94

DIGITAL SIGNAL
Digital Signal: Information can also be explained in the form of a digital signal. A digital signal can be explained with the help of following points:
A digital is a signal that has discrete values.
The signal will have value that is not continuous.

Question sources

Computer Science model exam 2 · Question 47

Question 70 | Q00707

Communication between a radio station and its listeners involves _____ transmission.

A. full-duplex

B. Half-duplex

C. Simplex

D. Automatic

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: Simplex

An ordinary broadcast radio station transmits to listeners who only receive. The one-direction relationship matches simplex. Evidence: PDF 5.

Why each choice is right or wrong

A - Incorrect: Full duplex requires both sides simultaneously transmitting (PDF 5.)

B - Incorrect: Half duplex requires alternate two-way transmission (PDF 5.)

C - Correct: Simplex explicitly has one sender and one receiver direction (PDF 5.)

D - Incorrect: Automatic is not one of the module transmission modes (PDF 5.)

Primary note proof

Study-note section NET06

> Simplex is one way, half duplex alternates directions, and full duplex permits simultaneous sending/receiving. A walkie-talkie is half duplex; a telephone conversation uses full duplex. Serial sends sequential bits; parallel uses multiple conductors and may suffer skew. Character-oriented asynchronous serial adds start/stop bits; synchronous framing uses ongoing clock recovery. Neither is synonymous with browser asynchronous requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 5 | lines 43-48

> • Simplex: Only one of the devices sends the data and the other one only receives the data (Unidirectional).
> • Half Duplex: both the stations can transmit as well as receive but not at the same time.
> • Full duplex mode: both stations can transmit and receive at the same time.

Question sources

Computer Science model exam 2 · Question 48

Question 71 | Q00708

Which of the following does not describe router functions

A. Broadcast forwarding

B. Internetwork communication

C. Packet filtering

D. Packet switching

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer A: Forwarding local broadcasts is not a normal router function

Routers forward IP packets between networks and separate broadcast domains. Accordingly internetwork communication and packet forwarding/switching describe routing, whereas ordinary local link broadcasts are not propagated across routers. Packet filtering can accompany the traffic-control function, so A is the normal-function exception, not a claim that no specially configured broadcast relay can exist.

Why each choice is right or wrong

A - Correct: Ordinary routers bound broadcast domains rather than forwarding local link broadcasts between them.

B - Incorrect: Forwarding IP packets between different networks is the defining router function.

C - Incorrect: Traffic filtering can be implemented alongside routing; it is consistent with router security-policy controls.

D - Incorrect: A router forwards individual packets to next hops, a network-layer packet-switching function.

Primary note proof

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Study-note section NET11

> A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and

demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device
used tointerconnect different types of network that
uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to
forwardpackets.
• The list of routing tables provides directions to
transfer the data to a particular network
destination -various paths to forward the data.
• A router analyzes a destination IP address of a
givenpacket and compares it with the routing
table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Data Communication and Computer Networking module | PDF page 13 | lines 2-8

Filters content by reading the MAC addresses of source and
destination.
• MAC to Segment number table
• Do not filter broadcast
• Data not destined for other network is prevented from
passing over the bridge
• Slower than repeater due to filtering

Question sources

Computer Science model exam 2 · Question 49

Question 72 | Q00709

Which OSI layer uses acknowledgments, sequencing and receiver flow control for end-to-end process delivery, as illustrated by TCP?

A. Layer 2

B. Layer 4

C. Layer 3

D. Layer 7

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Layer 4

Scope separates otherwise similar mechanisms. Reliable link procedures can acknowledge frames and control a single hop, whereas TCP uses sequence numbers, acknowledgments and windows between transport endpoints. The explicit end-to-end process-delivery requirement identifies layer 4. IP at layer 3 offers best-effort delivery, and application services build on the transport service rather than defining this TCP reliability machinery.

Why each choice is right or wrong

A - Incorrect: Layer 2 can provide similar mechanisms only for hop-local link delivery, outside the requested scope.

B - Correct: Layer 4 transport provides TCP’s end-to-end sequencing, acknowledgments and receiver flow control.

C - Incorrect: IP best-effort delivery does not provide the indicated ordered acknowledgment/retransmission service.

D - Incorrect: Application services can build reliability, but the standard link/transport mechanisms named here are not layer-7 duties.

Primary note proof

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.

• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 14 | lines 12-29

The Open Systems Interconnection (OSI) Model was developed by International Organization for Standardization (ISO). ISO is the organization, OSI is the model
It was developed to allow systems with different platforms (Hardware & software) to communicate with each other.
Now, it is a network model that defines the protocols for network communications.
It is a hierarchical model that groups its processes into seven (7) layers. (Top to Bottom)
1. Application Layer
2. Presentation Layer
3. Session Layer
4. Transport Layer
5. Network Layer
6. Data Link Layer
7. Physical Layer
Each layer has specific functions to perform and has to co-operate with the layers above and below it.

Question sources

Computer Science model exam 2 · Question 50

Question 73 | Q00710

What is the technique of combining multiple input signals or streams onto a shared transmission link?

A. Broadcast

B. Multicast

C. Multiplexing

D. demultiplexing

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer C: Multiplexing

Multiplexing combines input signals or streams so they share a link, using distinctions such as frequency, wavelength or time slots. Demultiplexing is the corresponding receiver operation that separates them. Broadcast and multicast instead describe recipient sets for a message. Link sharing and destination distribution therefore answer different questions, even though a network can use both mechanisms.

Why each choice is right or wrong

A - Incorrect: Broadcast distributes a message to all members of a broadcast domain rather than combining input streams.

B - Incorrect: Multicast distributes to a receiver group rather than combining inputs onto a link.

C - Correct: Multiplexing combines inputs onto the shared link.

D - Incorrect: Demultiplexing separates received multiplexed inputs; it is the reverse operation.

Primary note proof

Study-note section NET32

> Multiplexing combines multiple input streams onto a shared link; demultiplexing separates them at the receiver. It differs from broadcasting one message to many destinations. FDM assigns distinct frequency bands with suitable guard spacing; WDM assigns different optical wavelengths. TDM allocates time slots: synchronous TDM reserves recurrent slots even if an input is idle, while statistical TDM allocates according to demand and needs identification.

Study-note section NET30

> Flooding sends copies on eligible outgoing links with duplicate/lifetime controls; random walk picks a neighbor probabilistically. Unicast targets one destination, broadcast all in a scope, multicast a subscribed group and anycast one member sharing an address, normally nearest under routing metrics. Multi-destination packets can list multiple targets; this is not necessarily identical to broadcast. Group multicast trees use mechanisms such as reverse-path checks; Ethernet spanning tree is a different layer-2 mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 37-42

> MULTIPLEXING
> Whenever the bandwidth of a medium linking two devices is greater than the bandwidth needs of the devices, the link can be shared. Multiplexing is the set of techniques that allows the simultaneous transmission of multiple signals across a single data link.

Data Communication and Computer Networking module | PDF page 17 | lines 54-59

> ICMP- Internet Control Message Protocol: It is a signaling mechanism used to inform the sender about datagram problemsthat occur during transit. It is used by intermediate devices.
> IGMP- Internet Group Message Protocol: It is a mechanism that allows sending the same message to a group of recipients.

Question sources

Computer Science model exam 2 · Question 51

Question 74 | Q00912

How many layers are there in OSI reference model?

A. 7

B. 6

C. 5

D. 4

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: 7

OSI is explicitly a seven-layer hierarchy. Evidence: PDF 14, 17.

Why each choice is right or wrong

A - Correct: Seven is the stated count (PDF 14, 17.)

B - Incorrect: Six omits one stated layer (PDF 14, 17.)

C - Incorrect: Five is not the stated OSI count (PDF 14, 17.)

D - Incorrect: Four is the original TCP/IP count (PDF 14, 17.)

Primary note proof

Study-note section NET03

> A reference model divides responsibilities and interfaces; it is not itself executable networking software. OSI is a seven-layer reference model; TCP/IP describes an implemented protocol suite, usually taught with four layers. A five-layer teaching model separates physical and link functions. State the model before counting layers.

Study-note section NET19

> The four-layer TCP/IP model folds OSI layers 5-7 into application, retains transport, calls layer 3 Internet, and combines link/physical as network access. A model is a guide to functions, not a

guarantee that every application gets a reliable network. Distinguish end-to-end application service from its underlying process-to-process transport.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 12-29

The Open Systems Interconnection (OSI) Model was
developed by International Organization for Standardization
(ISO). ISO is the organization, OSI is the model
It was developed to allow systems with different platforms
(Hardware & software) to communicate with each other.
Now, it is a network model that defines the protocols for
network communications.
It is a hierarchical model that groups its processes into seven
(7) layers. (Top to Bottom)
1. Application Layer
2. Presentation Layer
3. Session Layer
4. Transport Layer
5. Network Layer
6. Data Link Layer
7. Physical Layer
Each layer has specific functions to perform and has to co-
operate with the layers above and below it.

Data Communication and Computer Networking module | PDF page 17 | lines 3-15

The structure TCP/IP model is very similar to the
structure of the OSI reference model. The OSI model has
seven layers where the TCP/IP model has four layers.
• The Application layer of TCP/IP model
corresponds to the Application Layer of Session,
Presentation & Application Layer of OSI model.
• The Transport layer of TCP/IP model corresponds
to theTransport Layer of OSI model
• The Network layer of TCP/IP model corresponds
to the Network Layer of OSI model
• The Host to network layer of TCP/IP model
correspondsto the Physical and Data link Layer of
OSI model.

Question sources

Computer Science Exit Exam · 2018 · Question 25

Question 75 | Q00921

Which one of the following ip address class uses first two octets for network addresses and last two for host addressing?

A. Class C

B. Class D

C. Class A

D. Class B

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer D: Class B

Two octets are sixteen bits. A historical Class B default /16 therefore uses the first two octets as network bits and the final two as host bits. Class A defaults to one network octet and Class C to three; Class D is multicast. Actual CIDR allocation can override the historical defaults.

Why each choice is right or wrong

A - Incorrect: Class C defaults to /24, using three network octets and one host octet.

B - Incorrect: Class D denotes multicast and does not use the described ordinary network/host division.

C - Incorrect: Class A defaults to /8, using one network octet and three host octets.

D - Correct: Class B defaults to /16, exactly the two-network-octet, two-host-octet division described.

Primary note proof

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Study-note section NET24

> A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science Exit Exam · 2018 · Question 34

Question 76 | Q00950

Which of the following OSI layers is NOT correctly matched to its corresponding data

Units (PDU)?

A. Application layer ---> Data

B. Transport layer ---> Segment

C. ¢.

Physical layer ---> Bit

D. Network layer ---> Frame

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Network layer ---> Frame

Network data units are packets, while frames are data-link units. The other options follow the described bit/segment and upper-data roles. Evidence: PDF 16, 15, 15, 14, 16.

Why each choice is right or wrong

A - Incorrect: Application handles user data

B - Incorrect: Transport explicitly forms segments

C - Incorrect: Physical transmits bits

D - Correct: Frame belongs to data link; network is packet

Primary note proof

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver

(Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.

• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Question sources

Computer Science Exit Exam · 2018 · Question 63

Question 77 | Q00959

Which of the following layers is responsible for encryption, translation and compression?

A. Presentation layer

B. Data link layer

C. .
Transport layer

D. Network layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: Presentation layer

The presentation paragraph explicitly names translation, encryption and compression together. Evidence: PDF 16, 15, 16, 15.

Why each choice is right or wrong

A - Correct: All three functions appear in the presentation definition (PDF 16, 15, 16, 15.)

B - Incorrect: Data link handles frames and link control (PDF 16, 15, 16, 15.)

C - Incorrect: Transport handles segmentation and process delivery (PDF 16, 15, 16, 15.)

D - Incorrect: Network handles addressing and routes (PDF 16, 15, 16, 15.)

Primary note proof

Study-note section NET19

OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.

• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize

devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Question sources

Computer Science Exit Exam · 2018 · Question 72

Question 78 | Q00962

Which service used to translate domain names to ip address?

A. HTTPS

B. DHCP

C. HTTP

D. DNS

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer D: DNS

DNS is explicitly the external naming service and Domain Name Service; the hosts-file passage contrasts it with stored host addresses. Evidence: PDF 18, 65, 28.

Why each choice is right or wrong

A - Incorrect: HTTPS is listed by port, not as naming service (PDF 18, 65, 28.)

B - Incorrect: DHCP is host configuration, not the named DNS service (PDF 18, 65, 28.)

C - Incorrect: HTTP is not named as the naming service (PDF 18, 65, 28.)

D - Correct: DNS matches the domain naming function (PDF 18, 65, 28.)

Primary note proof

Study-note section NET20

DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

/etc/hosts
The /etc/hosts file always contains the localhost IP
address, 127.0.0.1, which is
usedfor interprocess communication.
Never remove this line! Sometimes contains
addresses of additional hosts, whichcan be contacted
without using an external naming service such as
DNS (the Domain Name Server).

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the
combined session, presentation, and application layers in
the OSI model and many protocols are defined at this
layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science Exit Exam · 2018 · Question 75

Question 79 | Q00968

Which scenario is best suited for UDP rather than TCP protocol?

A. Live video streaming

B. Email transmission

C. Downloading the files

D. Database update

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A: Live video streaming

UDP is connectionless and omits flow/error control, fitting live streams where timely delivery matters more than repairing old data. Email/file/database transactions need the reliability contrast rather than that latency preference. Evidence: PDF 18, 18.

Why each choice is right or wrong

A - Correct: Live streaming fits the no-repair-delay tradeoff (PDF 18, 18.)

B - Incorrect: Mail transfer is an application for which reliable delivery is preferable (PDF 18, 18.)

C - Incorrect: File download generally needs intact reliable bytes (PDF 18, 18.)

D - Incorrect: Database updates require dependable ordered delivery (PDF 18, 18.)

Primary note proof

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Computer Science Exit Exam · 2018 · Question 81

Question 80 | Q00976

Assume that you have IPv4 addresses in binary notation 11000001 10000011 00011011 11111111 and you are requested to change into dotted decimal notation. What is the equivalent IP address in decimal dotted notation?

A. 192.174.27.255

B. 192.134.67.254

C. ¢.
193.130.28.254

D. 193.131.27.255

Hint

Sum the eight bit weights in each octet independently, then preserve their order in dotted-decimal notation.

Answer D: 193.131.27.255

Compute each octet: 11000001=128+64+1=193; 10000011=128+2+1=131; 00011011=16+8+2+1=27; 11111111=255. Evidence: PDF 19.

Why each choice is right or wrong

A - Incorrect: First and second octets disagree with 193 and 131 (PDF 19.)

B - Incorrect: 192,134,67,254 disagree with the independently converted octets 193,131,27,255 (PDF19).

C - Incorrect: Only first octet matches; 130,28,254 do not (PDF 19.)

D - Correct: 193.131.27.255 matches all four converted octets (PDF 19.)

Primary note proof

Study-note section NET22

Write IPv4 as four decimal octets 0-255, or as 32 binary bits. Use weights 128,64,32,16,8,4,2,1 within each octet. Thus 11000001 10000011 00011011 11111111 is 193.131.27.255. A value such as 257 is impossible. Syntactic validity differs from assignability: an address ending .0 or .255 can be a host under some prefixes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Question sources

Computer Science Exit Exam · 2018 · Question 89

Question 81 | Q00979

Which one of the following is disadvantage of circuit switching?

A. Predictable performance

B. Low latency

C. Guaranteed bandwidth

D. Limited scalability

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer D: Limited scalability due to reserved resources

Circuit switching reserves resources for an established path. NET31 explains that this offers predictable bandwidth but introduces setup, blocking and unused reservation during silence. Those costs constrain how many simultaneous circuits can be supported as demand grows, making limited scalability the disadvantage among the offered choices.

Why each choice is right or wrong

A - Incorrect: Predictable performance is a benefit of allocated circuit resources, not the disadvantage requested.

B - Incorrect: Low transfer latency after setup can be beneficial; the relevant costs include setup and resource reservation.

C - Incorrect: Guaranteed allocated bandwidth is an advantage, although reserving it consumes capacity during silence.

D - Correct: Finite reserved resources and blocking constrain simultaneous connections and therefore scalability as demand grows.

Primary note proof

Study-note section NET31

> Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 12-26

> Circuit Switching: In circuit switching, two
> communicating stations are connected by a dedicated

communication path which consists of intermediate nodes in the network and the links that connect these nodes.
It involved the following three distinct steps
• Circuit Establishment: To establish an end-to-end connection before any transfer of data.
• Data transfer: Transfer data is from the source to the destination. The data may be analog or digital, depending on the nature of the network. The connection is generally full-duplex.
• Circuit disconnect: Terminate connection at the end of data transfer. Signals must be propagated to deallocate the dedicated resources.

Data Communication and Computer Networking module | PDF page 20 | lines 27-36

Message Switching:
Packet Switching: Packet switching was designed to address the shortcomings of circuit switching in dealing with data communication. Unlike circuit switching where communication is continuous along a dedicated circuit, in packet switching, communication is discrete in form of packets.
The three switching modes to choose from are:
• Cut-through
• Store-and-forward
• Fragment-free

Question sources

Computer Science Exit Exam · 2018 · Question 92

Question 82 | Q00997

In which case must a dedicated communication path between the sender and receiver be established?

A. Circuit switching

B. Packet switching

C. A and B

D. None of the above

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: Circuit switching

A dedicated path is explicitly required by circuit switching. Packet communication is contrasted with that continuous dedicated circuit.

Why each choice is right or wrong

A - Correct: Circuit establishment creates the dedicated path

B - Incorrect: Packet communication is discrete rather than the dedicated circuit

C - Incorrect: Both incorrectly adds a requirement not inherent to packet switching

D - Incorrect: None excludes the explicitly dedicated circuit

Primary note proof

Study-note section NET31

> Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Study-note section NET31

> Datagram packets are forwarded independently. Virtual circuits establish forwarding state, but resource reservation is optional, not a necessary property of every virtual circuit. Message switching stores a complete message before forwarding, without a dedicated end-to-end path. Packet store-and-forward waits for the full packet/frame; cut-through starts after enough header arrives; fragment-free historically waits through the initial collision-sensitive portion. No routing algorithm guarantees that every packet always takes the geometrically shortest route.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 12-26

> Circuit Switching: In circuit switching, two
> communicating stations are connected by a dedicated
> communication path which consists of intermediate
> nodes in the network and the links that connect these
> nodes.
> It involved the following three distinct steps
> • Circuit Establishment: To establish an end-to-end
> connection before any transfer of data.
> • Data transfer: Transfer data is from the source to the
> destination. The data may be analog or digital,
> depending on the nature of the network. The
> connection is generally full-duplex.
> • Circuit disconnect: Terminate connection at the end

of data transfer. Signals must be propagated to deallocate the dedicated resources.

Data Communication and Computer Networking module | PDF page 20 | lines 27-36

Message Switching:
Packet Switching: Packet switching was designed to address the shortcomings of circuit switching in dealing with data communication. Unlike circuit switching where communication is continuous along a dedicated circuit, in packet switching, communication is discrete in form of packets.
The three switching modes to choose from are:
• Cut-through
• Store-and-forward
• Fragment-free

Question sources

National Exit Exam practice questions · Question 11

Computer Science Exit Exam practice collection · Question 151

Question 83 | Q00999

Which one of the following is a type of network topology in which all devices in the network are interconnected?

A. Star Topology

B. Ring Topology

C. Mesh Topology

D. Tree Topology

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer C: Mesh Topology

Interpret interconnected as every node directly connected to every other node: this is the module full-mesh definition. Without directly, all connected topologies would qualify. Evidence: PDF 23, 23, 23.

Why each choice is right or wrong

A - Incorrect: Star nodes connect through a central device (PDF 23, 23, 23.)

B - Incorrect: Ring nodes directly connect only to two neighbors (PDF 23, 23, 23.)

C - Correct: Full mesh explicitly connects every node to every other (PDF 23, 23, 23.)

D - Incorrect: Tree is listed but not defined as all-pairs direct connectivity (PDF 23, 23, 23.)

Primary note proof

Study-note section NET08

Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Study-note section NET08

Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 45-53

Mesh Topology: in this topology devices are connected with many redundant interconnections between network nodes. In a full mesh topology, every node has a connection to every other node in the network, which makes it the most expensive topology over all the other topologies. The number of cables grows fast as the number of nodes increases, and it can be calculated by using the general formula ((n (n - 1)) /2), where n is the number of nodes in the

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Data Communication and Computer Networking module | PDF page 23 | lines 35-44

> Ring Topology: in this topology, all devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it. Some of its advantages include, easy to install and reconfigure, less expensive (than mesh), and performance is even despite the number of users. Its cons include, break in the ring (such as a disabled station) can disable the entire network, and limitations on media and traffic (limitation on ring length and number of devices).

Question sources

National Exit Exam practice questions · Question 13

Computer Science Exit Exam practice collection · Question 152

Question 84 | Q01004

Which statements about IPv4 and IPv6 addresses are correct? Select all that apply.

A. IPv4 is 128-bit and IPv6 is 28-bit

B. IPv4 is physical address

C. IPv6 is logical address

D. IPv4 is 32-bit and IPv6 is 128-bit

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer C, D: IPv6 is logical address; IPv4 is 32-bit and IPv6 is 128-bit

IPv4 addresses contain 32 bits, and IPv6 addresses contain 128 bits. Both are logical network-layer addresses, so IPv6 being logical is compatible with the width comparison. Physical or MAC addressing belongs to the link layer. The corrected selection instruction allows both true statements to be selected while retaining the original factual comparisons and the two incorrect distractors.

Why each choice is right or wrong

A - Incorrect: IPv4 is 32 bits and IPv6 is 128 bits; neither printed width is correct.

B - Incorrect: IPv4 is network-layer logical addressing, not a physical/MAC address.

C - Correct: IPv6 addresses identify network-layer endpoints logically rather than physical link interfaces.

D - Correct: This gives the correct address lengths: 32 bits for IPv4 and 128 for IPv6.

Primary note proof

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Study-note section NET23

IPv6 has 128-bit hexadecimal addresses, flexible prefixes and no IPv4-style broadcast. Sources, not routers, fragment IPv6 packets. IPv6 does not automatically encrypt traffic or make authentication unnecessary. Compare IPv6's larger 128-bit address space and encryption support separately; specify the property being evaluated.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of
network addresses in IPv4 led to the development of IPv6.
Due to this, we need to use private IP address inside the
organization and translate it to public IP address using
NAT.

Question sources

National Exit Exam practice questions · Question 18

Computer Science Exit Exam practice collection · Question 154

Question 85 | Q01006

Which one of the data communication mediums is used for direct transmission?

A. Induced

B. Crosstalk

C. A and B

D. None of the above

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer D: Neither induced nor crosstalk names a transmission medium

The note identifies cables and unguided radio/microwave/infrared as transmission media. Crosstalk instead names interference from nearby channels in NET06, and induced alone does not name a medium in the note. Regardless of the vague direct-transmission phrase, none of A-C supplies a listed medium, so D is the supported exclusion response.

Why each choice is right or wrong

A - Incorrect: Induced is not a transmission-medium category established here.

B - Incorrect: Crosstalk is unwanted coupling/interference, not a medium for intentional direct transmission.

C - Incorrect: Combining two nonmedium terms does not supply the requested transmission medium.

D - Correct: Neither induced nor crosstalk names a transmission medium among the choices.

Primary note proof

Study-note section NET07

> Radio, microwave and infrared are unguided media. Ground waves follow Earth, sky waves can reflect from ionized layers, and line-of-sight paths require suitable visibility. Microwaves are radio waves in the broader physical classification. Infrared generally does not penetrate walls. Directionality depends on antennas: omnidirectional means broad azimuth coverage, not equal radiation in every 3-D direction. Legacy Ethernet straight-through cables join unlike MDI/MDI-X roles; crossover joins like roles, such as PC-PC or switch-switch. Auto-MDI-X can remove that requirement.

Study-note section NET06

> Attenuation reduces signal strength. Distortion changes shape, often because frequencies have different delays. Noise adds unwanted energy: thermal noise comes from random thermal motion, crosstalk leaks from nearby channels, and impulse noise is a short disturbance. Amplifiers amplify noise too; digital repeaters regenerate decisions but cannot reconstruct every corrupted bit.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 10 | lines 36-55

TRANSMISSION MEDIA
Transmission media is a means by which a communicationsignal is carried from one system to another
A transmission medium can be defined as anything that cancarry information from a source to a destination. The transmission medium is usually free space, metallic cable orfiber optic cable. Guided and Unguided (WIRELESS) Guided Transmission Media uses a cabling system that guidesthe data signals along a specific path. Out of these twisted-paircable, coaxial cable transport signals in the form of electricsignals and fiber-optic cable transport signals in the form oflight.
Unguided Transmission Media transport data without using a physical conductor. It uses wireless electromagnetic signals to send data. Three types of unguided media:
• Radio waves
• Micro waves
• Infrared.

Data Communication and Computer Networking module | PDF page 10 | lines 18-35

receiver.There are three causes of impairment:
• Attenuation
• Distortion
• Noise
Attenuation
Attenuation results in loss of energy due to distance. Whena signal travels through a medium, it loses some of its energy in overcoming the resistance of the medium. The electrical energy in the signal may convert to heat. To compensate for this loss, amplifiers are used to amplify thesignal.
Distortion
Distortion changes the shape of the signal.
Noise
Noise is any unwanted signal that is mixed or combined with the original signal during transmission. Due to noise the original signal is altered and signal received is not sameas the one sent.

Question sources

National Exit Exam practice questions · Question 20

Computer Science Exit Exam practice collection · Question 153

Question 86 | Q01160

Which OSI layer is responsible for process-to-process delivery and segmentation?

A. Network Layer

B. Transport Layer

C. Physical Layer

D. Data Link Layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Transport Layer

Transport supplies process-to-process delivery and segmentation. The network layer handles logical addressing and routing, the data-link layer handles hop delivery of frames, and the physical layer transports bits or symbols. Restoring the four readable source rows gives the transport option its proper position. Reliability remains protocol-dependent: identifying transport does not imply that every transport protocol guarantees delivery.

Why each choice is right or wrong

A - Incorrect: Network handles routing and logical addressing rather than process delivery.

B - Correct: Transport supplies process delivery and segmentation.

C - Incorrect: Physical transports signals and bits rather than process messages.

D - Incorrect: Data link supplies hop-local frame delivery rather than process-to-process delivery.

Primary note proof

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to

physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Question sources

Computer Science Exit Exam practice questions · Question 21

Question 87 | Q01161

Which IP version uses 32-bit addresses and corresponding 32-bit subnet masks?

A. IPv6

B. IPv4

C. Addressing

D. None of the above

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: IPv4

IPv4 uses 32-bit addresses, commonly written as four decimal octets, and its subnet mask selects a prefix within those 32 bits. IPv6 addresses instead contain 128 bits. The generic word addressing does not identify an IP version, and none of the above fails because the required version is listed.

Why each choice is right or wrong

A - Incorrect: IPv6 uses 128-bit addresses, not the required 32-bit addresses.

B - Correct: IPv4 addresses and their masks contain 32 bits.

C - Incorrect: Addressing is a general activity, not the IP version requested.

D - Incorrect: IPv4 is listed, so none of the above is false.

Primary note proof

Study-note section NET22

> Write IPv4 as four decimal octets 0-255, or as 32 binary bits. Use weights 128,64,32,16,8,4,2,1 within each octet. Thus 11000001 10000011 00011011 11111111 is 193.131.27.255. A value such as 257 is impossible. Syntactic validity differs from assignability: an address ending .0 or .255 can be a host under some prefixes.

Study-note section NET23

> IPv6 has 128-bit hexadecimal addresses, flexible prefixes and no IPv4-style broadcast. Sources, not routers, fragment IPv6 packets. IPv6 does not automatically encrypt traffic or make authentication unnecessary. Compare IPv6's larger 128-bit address space and encryption support separately; specify the property being evaluated.

Study-note section NET24

A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide's 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the Internet. Every host and router on the Internet has an IP address, which encodes its network number and host number.
• The combination is unique: in principle, no two machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination address fields of IP packets.
• An IP address does not refer to a host but it refers to a network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science Exit Exam practice questions · Question 22

Question 88 | Q01195

What is the main goal of data compression while it paths in different network layers? Its goal is to:

A. Securing data

B. Minimize the size of data

C. Select the best path of destination devices

D. All of the above

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Minimize the size of data

NET19 distinguishes reducing representation size from hiding content. Compression targets the amount of data needed to represent information, while encryption provides concealment and routing selects paths. Therefore minimizing size is the compression goal; combining it with security and path selection in the all option misattributes unrelated functions.

Why each choice is right or wrong

A - Incorrect: Securing or concealing content is an encryption/security function, not the primary goal of compression.

B - Correct: Compression reduces representation size, directly matching minimization of data size.

C - Incorrect: Path selection belongs to routing rather than the representation-size transformation performed by compression.

D - Incorrect: Only size reduction describes compression here; security and route selection are separate functions.

Primary note proof

Study-note section NET19

> OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET30

> Routing computes paths; forwarding chooses a next hop for an individual packet. Select the matching route with the longest prefix, then apply the routing system’s tie-breaking criteria. A default /0 matches when nothing more specific does. Static routes are configured; dynamic protocols exchange reachability. Distance vector uses neighbor estimates, while link state

distributes link information and computes shortest paths. Interior protocols operate within an autonomous system; BGP exchanges routes between systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Question sources

Computer Science model exam 2 · Question 16

Question 89 | Q01218

Which single option gives the complete set of listed protocols used by end users to retrieve or synchronize stored mailboxes?

A. POP3

B. IMAP

C. SMTP

D. Both POP3 and IMAP

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer D: Both POP3 and IMAP

POP3 retrieves stored messages, while IMAP accesses and synchronizes mailboxes held on the server. SMTP instead submits and relays messages between mail-system components. Since the question asks for the complete set of mailbox-access protocols, either POP3 or IMAP alone is incomplete. The combined choice correctly includes both without assigning mailbox retrieval to the mail transfer protocol.

Why each choice is right or wrong

A - Incorrect: POP3 retrieves messages but alone omits IMAP, so it is an incomplete selection.

B - Incorrect: IMAP accesses mailboxes but alone omits POP3, so it is incomplete.

C - Incorrect: SMTP submits and relays messages; it is not the end-user mailbox retrieval protocol described.

D - Correct: Both POP3 and IMAP provide the requested mailbox-access functions.

Primary note proof

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the
combined session, presentation, and application layers in
the OSI model and many protocols are defined at this
layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Question sources

Computer Science model exam 2 · Question 39

Question 90 | Q01229

The data link layer divides the stream of bits received from the physical layer into manageable data units called _____?

A. Bits

B. Frames

C. Packets

D. Segments

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Frames

The receive-side data-link description explicitly regroups physical-layer bits into frames. Evidence: PDF 15, 16, 15, 14.

Why each choice is right or wrong

A - Incorrect: Bits are the incoming physical units, not the regrouped unit (PDF 15, 16, 15, 14.)

B - Correct: Frames are explicitly the regrouped units (PDF 15, 16, 15, 14.)

C - Incorrect: Packets are network-layer units (PDF 15, 16, 15, 14.)

D - Incorrect: Segments are transport-layer units (PDF 15, 16, 15, 14.)

Primary note proof

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

> Data Link Layer
> • The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
> • Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver

side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into

packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Question sources

Computer Science model exam 2 · Question 50

Question 91 | Q01258

What is the maximum number of IP addresses that can be assigned to hosts on a local subnet that uses the 255.255.255.224 subnet mask?

A. 14

B. 32

C. 24

D. 30

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer D: 30 usable host addresses

The final mask octet 224 is 11100000, making the prefix /27 and leaving five host bits. There are 2^5=32 addresses in each block. For an ordinary broadcast subnet, one identifies the network and one the broadcast, leaving 30 assignable host addresses; total addresses must not be confused with usable hosts.

Why each choice is right or wrong

A - Incorrect: Fourteen usable hosts would correspond to /28, not this /27 mask.

B - Incorrect: Thirty-two is the total address count before reserving the network and broadcast addresses.

C - Incorrect: Twenty-four is not the usable count obtained from five host bits and two reservations.

D - Correct: Five host bits yield 32 total addresses minus two reserved addresses, leaving 30 hosts.

Primary note proof

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Study-note section NET24

> A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide's 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address, the address space is 2N because each bit can have twodifferent values (0 or 1) and N bits can have 2N values.
• IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science model exam 2 · Question 79

Question 92 | Q01270

Which one of the following is not LAN technologies?

A. Token Ring

B. Ethernet

C. X.25 or Frame Relay

D. None

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: X.25 or Frame Relay

The module explicitly contrasts LAN Ethernet/Token Ring with WAN X.25/Frame Relay. Evidence: PDF 26.

Why each choice is right or wrong

A - Incorrect: Token Ring is explicitly a LAN technology (PDF 26.)

B - Incorrect: Ethernet is explicitly a LAN technology (PDF 26.)

C - Correct: X.25 and Frame Relay are explicitly WAN technologies (PDF 26.)

D - Incorrect: None fails because the WAN pair is present (PDF 26.)

Primary note proof

Study-note section NET-S05

> Ethernet and Token Ring are traditional LAN technologies. X.25 and Frame Relay are traditional WAN technologies. TCP/IP can run over different network-access technologies; its Internet-layer packet service does not require one particular link medium or frame format.

Study-note section NET31

> Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 26 | lines 14-28

> medium. TCP/IP was designed to be independent
> of the network access method, frame format, and
> medium. In this way, TCP/IP can be used to

connect differing network types. These include LAN technologies such as Ethernet and Token Ring and WAN technologies such as X.25 and Frame Relay. Independence from any specific network technology gives TCP/IP the ability to be adapted to new technologies such as Asynchronous Transfer Mode (ATM).
Network Access layer uses a physical address to identify hosts and to deliver data.
The Network Access layer PDU is called a frame. It contains the IP packet as well as a protocol header and trailer from this layer.

Question sources

Computer Science model exam 2 · Question 91

Question 93 | Q01276

Which internet layer address is responsible for providing diagnostic functions and reporting errors or conditions regarding the delivery of IP packets?

A. Internet Protocol (IP)

B. Address Resolution Protocol (ARP)

C. Internet Control Message Protocol (ICMP)

D. Internet Group Message Protocol (IGMP)

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: Internet Control Message Protocol (ICMP)

ICMP reports transit problems and supports destination-unreachable, echo and timeout diagnostics. The stem calls it an address, but asks for a protocol by its functions. Evidence: PDF 17, 27, 17, 17.

Why each choice is right or wrong

A - Incorrect: IP supplies best-effort datagram delivery (PDF 17, 27, 17, 17.)

B - Incorrect: ARP resolves IP to hardware address (PDF 17, 27, 17, 17.)

C - Correct: ICMP explicitly provides error/diagnostic messages (PDF 17, 27, 17, 17.)

D - Incorrect: IGMP supports group recipients rather than these transit diagnostics (PDF 17, 27, 17, 17.)

Primary note proof

Study-note section NET20

ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target's MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 54-59

ICMP- Internet Control Message Protocol: It is a
signaling mechanism used to inform the sender about
datagram problemsthat occur during transit. It is used by
intermediate devices.
IGMP- Internet Group Message Protocol: It is a mechanism
that allows sending the same message to a group of recipients.

Data Communication and Computer Networking module | PDF page 27 | lines 5-31

Internet Control Message Protocol (ICMP):
works at the Network layer and is used by IP for
many different services. ICMP is a management
protocol and messagingservice provider for IP. The
following are some common events and messages
that ICMP relates to:
Destination Unreachable: If a router can‘t send an
IP datagram any further, it uses ICMP to send a
message back to the sender, advising it of the
situation.
Buffer Full: If a router‘s memory buffer for
receiving incoming datagrams is full, it will use
ICMP to send out this message until the congestion
abates.
Hops: Each IP datagram is allotted a certain number
of routers, called hops, to pass through. If it reaches
its limitof hops before arriving at its destination, the
last router to receive that datagram deletes it. The

executioner router then uses ICMP to send an obituary message, informing the sending machine of the demise of its datagram.
Ping (Packet Internet Groper) uses ICMP echo messages to check the physical and logical connectivity of machineson a network.
Traceroute Using ICMP timeouts, Traceroute is used to discover the path a packet takes as it traverses an Internetwork.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to resolve thephysical address of a device on a network, where its logical address is known.
Physical address is the 48 bit address that is imprinted on theNIC or LAN card, Logical address is the Internet Address orcommonly known as IP address that is used to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is used bya device on the network to find its Internet address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 17 | lines 22-38

Network Layer or IP
Also called as the Internetwork Layer (IP). It holds the IP protocol which is a network layer protocol and is responsiblefor source to destination transmission of data.
The Internetworking Protocol (IP) is a connection-less & unreliable protocol.
It is a best effort delivery service. i.e. there is no error checking in IP, it simply sends the data and relies on its underlying layers to get the data transmitted to the destination.Even if it is unreliable IP cannot be considered weak and useless; since it provides only the functionality that is requiredfor transmitting data thereby giving maximum efficiency.
Since there is no mechanism of error detection or correction in IP, there will be no delay introduced on a medium where thereis no error at all.

Question sources

Computer Science model exam 2 · Question 97

Question 94 | Q01277

Hana divides a 172.16.0.0/16 parent into equal /19 broadcast subnets. Counting every subnet, which option correctly gives the subnet count and usable hosts per subnet?

A. 7 subnets and 30 valid hosts

B. 8 subnets and 8190 usable hosts in each subnet

C. 8 subnets and 2046 valid hosts in each subnet

D. 7 subnets and 2046 valid hosts in each subnet

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: 8 subnets and 8190 usable hosts in each subnet

The /16-to-/19 subdivision borrows three bits, so it creates eight equal subnets. Each /19 leaves thirteen host bits, producing 8192 total addresses and 8190 usable hosts after network and broadcast reservations. Modern counting includes every subnet. The option changes only the obsolete subnet count; the calculation still requires distinguishing child subnet count from host capacity.

Why each choice is right or wrong

A - Incorrect: Neither seven subnets nor thirty usable hosts follows from /19; thirty hosts would require /27.

B - Correct: Three subnet bits give eight subnets and thirteen host bits give 8190 usable hosts.

C - Incorrect: Eight subnets is correct for the stated /16 parent, but a /19 has 8190 usable hosts rather than 2046.

D - Incorrect: Seven lacks a stated counting basis, and 2046 hosts corresponds to /21 rather than /19.

Primary note proof

Study-note section NET26

> To split parent /a into equal /p blocks, borrow b=p-a bits and obtain 2^b subnets. Do not subtract two subnets under modern practice. For a Class B /16 parent split to /19, b=3 gives eight subnets, each with 8192 addresses and 8190 conventional hosts. A /19 alone does not state its parent.

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet’s increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the Internet. Every host and router on the Internet has an IP address, which encodes its network number and host number.
• The combination is unique: in principle, no two machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination address fields of IP packets.
• An IP address does not refer to a host but it refers to a network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address, the address space is 2N because each bit can have twodifferent values (0 or 1) and N bits can have 2N values.
• IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex.

255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science model exam 2 · Question 98

Question 95 | Q01494

A switch-based LAN is more efficient than a hub-based LAN because:

A. A switch can connect two different LANs together

B. A switch allows multiple computers to send at the same time

C. A hub uses reservation-based MAC protocols

D. A switch uses contention-based MAC protocols

E. A switch is easier to implement than a hub.

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer B: A switch allows multiple computers to send at the same time

Switches direct frames toward the MAC recipient instead of copying all traffic to every port. Independent point-to-point communication permits multiple simultaneous exchanges on suitable separate ports. Evidence: PDF 12, 12.

Why each choice is right or wrong

A - Incorrect: Connecting LANs is not the efficiency distinction in the switch/hub comparison (PDF 12, 12.)

B - Correct: Selective independent port communication explains the improvement (PDF 12, 12.)

C - Incorrect: The module says hubs repeat/broadcast, not reservation-based MAC (PDF 12, 12.)

D - Incorrect: A switch is not defined by a contention-based MAC distinction from hubs (PDF 12, 12.)

E - Incorrect: The module calls switches more intelligent and expensive, not easier to implement (PDF 12, 12.)

Primary note proof

Study-note section NET10

A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does

not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET10

Each switch port separates collision domains; full-duplex point-to-point Ethernet has no collisions and does not use CSMA/CD. Switching permits independent conversations simultaneously. Broadcasts still reach the VLAN unless another control filters them. A layer-3 switch also routes, so device marketing names do not override the function being examined.

Study-note section NET09

A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets
across anetwork.
Unlike a hub which simply replicates what it receives on
oneport onto all the other ports, a switch keeps a record
of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the
packetdirectly to the recipient device by looking up the
MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is
normallyused to create connections between stations in
a physical startopology.
HUB: Hubs are commonly used to connect segments
(devices)of a LAN. Hubs don‘t filter information; instead
broadcast incoming packets to all computers connected to
it. A hub contains multiple ports which support 8, 12 or
24 RJ-45 ports. Two types: Passive hub and Active hub. It
is used in star or ring topology.

Question sources

Mock exam 1 · Question 10

Question 96 | Q01496

Compare circuit switching versus datagram packet switching. An advantage of circuit switching is it:

A. Allows end hosts to use different line rates.

B. Can guarantee the performance delivered to end hosts.

C. Establishes a connection before sending data.

D. Sends packets instead of signals.

E. Is non-blocking.

F. Is efficient when the amount of traffic to be sent by end hosts varies over time.

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer B: Reserved resources can guarantee performance

The reserved-resource circuit supports predictable allocated bandwidth and performance, the advantage contrasted with datagram links shared statistically. Establishing a connection is a procedural characteristic and imposes setup cost rather than itself guaranteeing a benefit. The note explicitly identifies blocking and wasted reservation during silence, ruling out the nonblocking and variable-traffic-efficiency claims.

Why each choice is right or wrong

A - Incorrect: Different endpoint line rates are not the reserved-circuit advantage established in the note.

B - Correct: Reserved resources provide predictable allocated bandwidth and can guarantee the agreed performance to endpoints.

C - Incorrect: Setup is a characteristic of circuit operation, but is not itself the performance advantage requested.

D - Incorrect: Circuit switching establishes reserved resources; packet-versus-signal wording does not describe its advantage.

E - Incorrect: NET31 explicitly identifies blocking as a circuit-switching tradeoff.

F - Incorrect: Reservations can be wasted during silence, making highly variable traffic less efficient on fixed circuits.

Primary note proof

Study-note section NET31

> Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into

packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Study-note section NET31

Datagram packets are forwarded independently. Virtual circuits establish forwarding state, but resource reservation is optional, not a necessary property of every virtual circuit. Message switching stores a complete message before forwarding, without a dedicated end-to-end path. Packet store-and-forward waits for the full packet/frame; cut-through starts after enough header arrives; fragment-free historically waits through the initial collision-sensitive portion. No routing algorithm guarantees that every packet always takes the geometrically shortest route.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 12-26

Circuit Switching: In circuit switching, two communicating stations are connected by a dedicated communication path which consists of intermediate nodes in the network and the links that connect these nodes.
It involved the following three distinct steps
• Circuit Establishment: To establish an end-to-end connection before any transfer of data.
• Data transfer: Transfer data is from the source to the destination. The data may be analog or digital, depending on the nature of the network. The connection is generally full-duplex.
• Circuit disconnect: Terminate connection at the end of data transfer. Signals must be propagated to deallocate the dedicated resources.

Data Communication and Computer Networking module | PDF page 20 | lines 27-36

Message Switching:
Packet Switching: Packet switching was designed to address the shortcomings of circuit switching in dealing with data communication. Unlike circuit switching where communication is continuous along a dedicated circuit, in packet switching, communication is discrete in form of packets.
The three switching modes to choose from are:
• Cut-through
• Store-and-forward
• Fragment-free

Question sources

Mock exam 1 · Question 12

Question 97 | Q01521

A _____ is a system in which multiple computers are connected to each other to share information and resources.

A. Hardware

B. Software

C. Number system

D. Computer network

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: Computer network

The module defines a network as connected computers sharing resources and exchanging information. Evidence: PDF 22.

Why each choice is right or wrong

A - Incorrect: Hardware alone need not connect computers for sharing (PDF 22.)

B - Incorrect: Software alone is not the system of connected devices (PDF 22.)

C - Incorrect: A number system is a representation scheme, not shared computers (PDF 22.)

D - Correct: Computer network matches the supplied definition (PDF 22.)

Primary note proof

Study-note section NET-S04

> A computer network connects computers or other devices so they can exchange information and share resources such as files, storage and printers. The interconnection uses both hardware and software; neither hardware alone nor software alone is the definition of the whole network.

Study-note section NET02

> An intranet serves an organization internally; an extranet exposes selected services to approved outsiders. Neither term alone specifies geographic size. A client requests a service; a server supplies it. Peer systems may perform both roles. Dedicated servers concentrate on service workloads but can still be clients of DNS, backup or update services. A ten-user cutoff is a teaching guideline, not a technical law. Centralization simplifies administration but creates dependencies; peer networks still need security.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 2-19

A network can be defined as two or more computers connected together in such a way that they can share resources. The primary purpose of a network is to share resources, and a resource could be:
• a file,
• a folder,
• a printer,
• a disk drive, or just about anything else that exists on a computer.
Therefore, a computer network is simply a collection of computers or other hardware devices that are connected together, either physically or logically, using special hardware and software, to allow them to exchange information and cooperate. Networking is the term that describes the processes involved in designing, implementing, upgrading, managing and otherwise working with networks and network technologies.

Question sources

Data Communication & Networking practice questions · Question 1

Question 98 | Q01522

NIC stands for?

A. Network Internal Card

B. Network Interconnect Card

C. Network Interface Card

D. Network Interface Component

Hint

Find the full expansion in the terminology note and compare each word rather than only matching initials.

Answer C: Network Interface Card

The star cabling paragraph explicitly expands NIC as Network Interface Card. Evidence: PDF 23.

Why each choice is right or wrong

A - Incorrect: Internal differs from the printed Interface (PDF 23.)

B - Incorrect: Interconnect differs from the printed Interface (PDF 23.)

C - Correct: Exact expansion is Network Interface Card (PDF 23.)

D - Incorrect: Component differs from the printed Card (PDF 23.)

Primary note proof

Study-note section NET-S04

> NIC expands to Network Interface Card. It is the host's network interface, providing the connection through which the host sends and receives network signals. A cable connector attaches the medium to that interface; it is distinct from an intermediary switch or router.

Study-note section NET09

> A NIC supplies a computer's network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

> Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Question sources

Data Communication & Networking practice questions · Question 2

Question 99 | Q01523

A term that refers to the way in which the nodes of a network are linked together.

A. network

B. topology

C. connection

D. interconnectivity

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B: topology

Topology is explicitly the shape/layout and relation among links and devices, matching the way nodes are linked. Evidence: PDF 22, 11.

Why each choice is right or wrong

A - Incorrect: Network is the collection of devices, not the specific arrangement term (PDF 22, 11.)

B - Correct: Topology is the defined arrangement term (PDF 22, 11.)

C - Incorrect: Connection is a generic link rather than the named layout concept (PDF 22, 11.)

D - Incorrect: Interconnectivity is the organizing attribute, while topology names its layout (PDF 22, 11.)

Primary note proof

Study-note section NET08

> Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Study-note section NET08

> Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 45-52

> Inter-connectivity: components of a network,
> including end devices and interconnecting devices,
> can be connected to each other differently in some
> fashion. By connectedness we mean either logically,
> physically or both ways. Network topology refers to
> the shape of a network, or the network‘s layout. It is
> the geometric representation of the relationship of all
> the links and linking devices to one another. There are

Data Communication and Computer Networking module | PDF page 11 | lines 34-51

> NETWORK TOPOLOGY
> A network topology is a description of the layout of the region

or area covered by that network.
There are two types of connections. They are: point-to-point and multi-point.
Point-to-point connections provide a direct link between two devices.
Multi-point connections provide a link between three or more devices on a network.
There are seven basic topologies in the study of network topology:
• Point-to-point topology,
• Bus (point-to-multipoint) topology,
• Ring topology,
• Star topology,
• Hybrid topology,
• Mesh topology and,
• Tree topology.

Question sources

Data Communication & Networking practice questions · Question 3

Question 100 | Q01524

A topology that involves Tokens.

A. Star

B. Bus

C. Ring

D. Daisy Chaining

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer C: Ring in the conventional Token Ring association

NET08 associates token passing with Token Ring, so ring is the intended topology among these choices. The answer is an association rather than a universal claim: token passing is an access method and not every ring must use it. None of the other topology names inherently establishes token access.

Why each choice is right or wrong

A - Incorrect: A star describes central connections and does not itself specify token access.

B - Incorrect: A bus describes a shared backbone; token access is not inherent in that layout.

C - Correct: Token Ring is the conventional association, with the caveat that tokens are an access method.

D - Incorrect: Daisy chaining describes successive connections and does not by itself establish token passing.

Primary note proof

Study-note section NET08

Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Study-note section NET14

Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 21 | lines 6-26

MEDIUM ACCESS CONTROL
The problem of controlling the access to the medium is similarto the rules of speaking in a meeting (the right to speak isupheld; two people do not speak at the same time; do notinterrupt each other; do not monopolize the discussion; ...) Three categories of multiple access protocols
Random Access Protocols - try your best like taxis do
• MA - Multiple Access
• CSMA - Carrier Sense MA
• CSMA/CD - CSMA with Collision Detection
• CSMA/CA - CSMA with Collision AvoidanceControlled-Access Protocols - get permission
• Reservation
• Polling
• Token Passing
Channelization Protocols - simultaneous use
• FDMA - Frequency-Division MA
• TDMA - Time-Division MA
• CDMA - Code-Division MA

Data Communication and Computer Networking module | PDF page 23 | lines 35-44

> Ring Topology: in this topology, all devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it. Some of its advantages include, easy to install and reconfigure, less expensive (than mesh), and performance is even despite the number of users. Its cons include, break in the ring (such as a disabled station) can disable the entire network, and limitations on media and traffic (limitation on ring length and number of devices).

Question sources

Data Communication & Networking practice questions · Question 4

Question 101 | Q01525

Physical or logical arrangement of network is _____

A. Topology

B. Routing

C. Networking

D. Control

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: Topology

Physical or logical relations among links and devices form the network topology; routing chooses paths over that arrangement. Evidence: PDF 22, 19.

Why each choice is right or wrong

A - Correct: Topology is explicitly the arrangement/layout

B - Incorrect: Routing chooses a packet path, not the whole arrangement

C - Incorrect: Networking includes designing and managing networks broadly

D - Incorrect: Control does not name the defined arrangement

Primary note proof

Study-note section NET08

> Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Study-note section NET30

Routing computes paths; forwarding chooses a next hop for an individual packet. Select the matching route with the longest prefix, then apply the routing system's tie-breaking criteria. A default /0 matches when nothing more specific does. Static routes are configured; dynamic protocols exchange reachability. Distance vector uses neighbor estimates, while link state distributes link information and computes shortest paths. Interior protocols operate within an autonomous system; BGP exchanges routes between systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 45-52

Inter-connectivity: components of a network,
including end devices and interconnecting devices,
can be connected to each other differently in some
fashion. By connectedness we mean either logically,
physically or both ways. Network topology refers to
the shape of a network, or the network‘s layout. It is
the geometric representation of the relationship of all
the links and linking devices to one another. There are

Data Communication and Computer Networking module | PDF page 19 | lines 52-64

ROUTING AND ROUTING PROTOCOLS
Routing is the act of moving information across an
internetwork from a source to a destination. At the heart of
such protocol is the routing algorithm that determines the
path for a packet.
• Routing requires a host or a router to have a
routingtable which is constructed by the routing
algorithm.
• There are wo kinds of routing algorithms (routing
tables): nonadaptive (static) and adaptive (dynamic)
• And two kinds of protocols
o Interior routing protocols
o Exterior routing protocols

Question sources

Data Communication & Networking practice questions · Question 5

Question 102 | Q01526

Which network topology requires a central controller or hub?

A. Star

B. Mesh

C. Ring

D. Bus

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer A: Star

Star explicitly connects each node to a central hub or switch. Mesh, ring and bus have different connection structures. Evidence: PDF 23, 23, 23, 23.

Why each choice is right or wrong

A - Correct: Central hub/switch is the defining star feature (PDF 23, 23, 23, 23.)

B - Incorrect: Full mesh uses all-pairs links (PDF 23, 23, 23, 23.)

C - Incorrect: Ring connects two neighbors per node (PDF 23, 23, 23, 23.)

D - Incorrect: Bus uses a common backbone rather than a controller (PDF 23, 23, 23, 23.)

Primary note proof

Study-note section NET08

> Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Study-note section NET08

> A full undirected mesh of n nodes needs n(n-1)/2 links and n-1 interfaces per node. For 50 nodes: 50×49/2=1225 links. Redundant paths improve fault tolerance at greater cable and interface cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

> Star Topology: in this topology, each node is
> connected directly to a central device called a hub or
> a switch. Data on a star network passes through the
> central device (switch) before continuing to its
> destination. The central device manages and controls

all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Data Communication and Computer Networking module | PDF page 23 | lines 45-53

Mesh Topology: in this topology devices are connected with many redundant interconnections between network nodes. In a full mesh topology, every node has a connection to every other node in the network, which makes it the most expensive topology over all the other topologies. The number of cables grows fast as the number of nodes increases, and it can be calculated by using the general formula ((n (n - 1)) /2), where n is the number of nodes in the

Data Communication and Computer Networking module | PDF page 23 | lines 35-44

Ring Topology: in this topology, all devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it. Some of its advantages include, easy to install and reconfigure, less expensive (than mesh), and performance is even despite the number of users. Its cons include, break in the ring (such as a disabled station) can disable the entire network, and limitations on media and traffic (limitation on ring length and number of devices).

Data Communication and Computer Networking module | PDF page 23 | lines 4-18

Bus Topology: in this topology all devices are connected to a central cable, called the bus or backbone, which is terminated at its ends. The purpose of the terminators is to stop the signal from bouncing, thereby clearing the cable so that other computers can send data. Message transmitted along the Bus is visible to all computers connected to the backbone cable. As the message arrives at each workstation, the workstation checks the destination address contained in the message either to process or

drop the packet if it matches or not respectively. Its advantages are, ease of installation and less amount of cable requirement. Its main drawback is, the entire network will be shut down if there is a break in the main cable.

Question sources

Data Communication & Networking practice questions · Question 6

Question 103 | Q01527

_____ topology requires a multipoint connection.

A. Star

B. Mesh

C. Ring

D. Bus

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer D: Bus uses a shared multipoint backbone

A multipoint connection shares one medium among endpoints. NET08 describes bus as a shared backbone, while a full mesh consists of separate point-to-point links. Ordinary star and ring link layouts connect endpoints through separate adjacent links rather than requiring a common shared bus. D is therefore the topology matching multipoint sharing.

Why each choice is right or wrong

A - Incorrect: A star connects endpoints through a central device using individual edge links rather than requiring a shared backbone.

B - Incorrect: A full mesh consists of point-to-point links, explicitly correcting the old contradictory multipoint claim.

C - Incorrect: A ring forms successive neighboring links, not the common shared backbone requested.

D - Correct: A bus shares one backbone among multiple endpoints, matching a multipoint connection.

Primary note proof

Study-note section NET08

Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Study-note section NET08

Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 11 | lines 34-51

NETWORK TOPOLOGY
A network topology is a description of the layout of the region or area covered by that network.
There are two types of connections. They are: point-to-point and multi-point.
Point-to-point connections provide a direct link between two devices.
Multi-point connections provide a link between three or more devices on a network.
There are seven basic topologies in the study of network topology:
• Point-to-point topology,
• Bus (point-to-multipoint) topology,
• Ring topology,
• Star topology,
• Hybrid topology,
• Mesh topology and,
• Tree topology.

Data Communication and Computer Networking module | PDF page 23 | lines 4-18

Bus Topology: in this topology all devices are connected to a central cable, called the bus or backbone, which is terminated at its ends. The purpose of the terminators is to stop the signal from bouncing, thereby clearing the cable so that other computers can send data. Message transmitted along the Bus is visible to all computers connected to the backbone cable. As the message arrives at each workstation, the workstation checks the destination address contained in the message either to process or drop the packet if it matches or not respectively. Its advantages are, ease of installation and less amount

of cable requirement. Its main drawback is, the entire network will be shut down if there is a break in the main cable.

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Data Communication and Computer Networking module | PDF page 23 | lines 45-53

Mesh Topology: in this topology devices are connected with many redundant interconnections between network nodes. In a full mesh topology, every node has a connection to every other node in the network, which makes it the most expensive topology over all the other topologies. The number of cables grows fast as the number of nodes increases, and it can be calculated by using the general formula ((n (n - 1)) /2), where n is the number of nodes in the

Question sources

Data Communication & Networking practice questions · Question 7

Question 104 | Q01528

Data communication system spanning states, countries, or the whole world is _____

A. LAN

B. WAN

C. MAN

D. PAN

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B: WAN

WAN explicitly spans multiple cities, countries, continents or the world. Evidence: PDF 22.

Why each choice is right or wrong

A - Incorrect: LAN is building/local organization span (PDF 22.)

B - Correct: WAN matches states/countries/world scale (PDF 22.)

C - Incorrect: MAN covers a city (PDF 22.)

D - Incorrect: PAN covers a few meters (PDF 22.)

Primary note proof

Study-note section NET02

> PAN connects personal devices, such as Bluetooth accessories. LAN covers a local building or campus; MAN spans a metropolitan region; WAN connects distant regions. Device count alone does not determine the category. The Internet is a global network of networks, commonly the largest example of a WAN.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 26-44

> Personal Area Network (PAN): is a network may be spanned across a given table with distances between the devices not more than few meters. The technology used to interconnect the devices could be a Bluetooth. These networks are called Personal Area Networks, since the devices interconnected in these networks are belongs to a single person.
> Local Area Network (LAN): is a network that may span across a building, or across several buildings within a single organization, by using intermediate devices, like switches and/or hubs, to interconnect devices in all floors.
> Metropolitan Area Network (MAN): is a network that may span across a whole city interconnecting several buildings and organizations.
> Wide Area Network (WAN): is a network that may span across multiple cities, or an entire country, or an entire continent or it may even cover the whole world. For example, an Internet is one example of WAN.

Question sources

Data Communication & Networking practice questions · Question 8

Question 105 | Q01529

Data communication system within a building or campus is_____

A. LAN

B. WAN

C. MAN

D. PAN

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: LAN

A LAN can span a building or several buildings within an organization, fitting the intended single-campus scale. Evidence: PDF 22.

Why each choice is right or wrong

A - Correct: LAN matches the described building/campus extent (PDF 22.)

B - Incorrect: WAN spans cities or beyond (PDF 22.)

C - Incorrect: MAN spans a city (PDF 22.)

D - Incorrect: PAN is a few-meter personal range (PDF 22.)

Primary note proof

Study-note section NET02

> PAN connects personal devices, such as Bluetooth accessories. LAN covers a local building or campus; MAN spans a metropolitan region; WAN connects distant regions. Device count alone does not determine the category. The Internet is a global network of networks, commonly the largest example of a WAN.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 26-44

> Personal Area Network (PAN): is a network may be spanned across a given table with distances between the devices not more than few meters. The technology used to interconnect the devices could be a Bluetooth. These networks are called Personal Area Networks, since the devices interconnected in these networks are

belongs to a single person.
Local Area Network (LAN): is a network that may span across a building, or across several buildings within a single organization, by using intermediate devices, like switches and/or hubs, to interconnect devices in all floors.
Metropolitan Area Network (MAN): is a network that may span across a whole city interconnecting several buildings and organizations.
Wide Area Network (WAN): is a network that may span across multiple cities, or an entire country, or an entire continent or it may even cover the whole world. For example, an Internet is one example of WAN.

Question sources

Data Communication & Networking practice questions · Question 9

Question 106 | Q01530

Bluetooth is an example of _____

A. personal area network

B. local area network

C. virtual private network

D. wide area network

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: personal area network

The PAN definition explicitly gives Bluetooth as the interconnection technology for a few-meter personal network. Evidence: PDF 22.

Why each choice is right or wrong

A - Correct: PAN is the explicit Bluetooth example (PDF 22.)

B - Incorrect: LAN is building scale, not the cited Bluetooth example (PDF 22.)

C - Incorrect: VPN is not the geographic class in the example (PDF 22.)

D - Incorrect: WAN is much wider than the personal example (PDF 22.)

Primary note proof

Study-note section NET02

PAN connects personal devices, such as Bluetooth accessories. LAN covers a local building or campus; MAN spans a metropolitan region; WAN connects distant regions. Device count alone does not determine the category. The Internet is a global network of networks, commonly the largest example of a WAN.

Study-note section NET11

A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 26-44

Personal Area Network (PAN): is a network may be spanned across a given table with distances between the devices not more than few meters. The technology used to interconnect the devices could be a Bluetooth. These networks are called Personal Area Networks, since the devices interconnected in these networks are belongs to a single person.
Local Area Network (LAN): is a network that may span across a building, or across several buildings within a single organization, by using intermediate devices, like switches and/or hubs, to interconnect devices in all floors.
Metropolitan Area Network (MAN): is a network that may span across a whole city interconnecting several buildings and organizations.
Wide Area Network (WAN): is a network that may span across multiple cities, or an entire country, or an entire continent or it may even cover the whole world. For example, an Internet is one example of WAN.

Question sources

Data Communication & Networking practice questions · Question 10

Question 107 | Q01531

A _____ is a device that forwards packets between networks by processing the routing information included in the packet.

A. bridge

B. firewall

C. router

D. hub

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer C: router

Router forwarding explicitly compares the destination IP to routing table entries to choose the next path between networks. Evidence: PDF 12, 12, 12, 48.

Why each choice is right or wrong

A - Incorrect: Bridge filters same-protocol LAN traffic using MAC (PDF 12, 12, 12, 48.)

B - Incorrect: A firewall permits/denies access; that is not the defining routing-table function (PDF 12, 12, 12, 48.)

C - Correct: Router matches the stated destination-IP routing procedure (PDF 12, 12, 12, 48.)

D - Incorrect: Hub repeats/broadcasts without destination-route analysis (PDF 12, 12, 12, 48.)

Primary note proof

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Study-note section NET09

> A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does

not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device
used tointerconnect different types of network that
uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to
forwardpackets.
• The list of routing tables provides directions to
transfer the data to a particular network
destination -various paths to forward the data.
• A router analyzes a destination IP address of a
givenpacket and compares it with the routing
table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 12 | lines 60-61

BRIDGE: Connects two LANS having the same protocol -
(e.g. Ethernet or Token ring).

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is
normallyused to create connections between stations in
a physical startopology.
HUB: Hubs are commonly used to connect segments
(devices)of a LAN. Hubs don‘t filter information; instead
broadcast incoming packets to all computers connected to
it. A hub contains multiple ports which support 8, 12 or
24 RJ-45 ports. Two types: Passive hub and Active hub. It
is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local
network to a single IP address. This address is often used
by the router that connects the computers to the Internet.
The router can be connected to a DSL modem, cable
modem, T1 line, or even a dial-up modem. When other
computers on the Internet attempt to access computers

within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Question sources

Data Communication & Networking practice questions · Question 11

Question 108 | Q01532

The physical layer is concerned with _____

A. bit-by-bit delivery

B. process to process delivery

C. application to application delivery

D. port to port delivery

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: bit-by-bit delivery

Physical-layer standardized interfaces deliver bit-by-bit or symbol-by-symbol; processes and ports are transport concerns. Evidence: PDF 14, 16.

Why each choice is right or wrong

A - Correct: Bit-by-bit delivery is explicit (PDF 14, 16.)

B - Incorrect: Process delivery is transport (PDF 14, 16.)

C - Incorrect: Applications communicate through upper and transport services, not the physical bit interface (PDF 14, 16.)

D - Incorrect: Ports identify transport processes (PDF 14, 16.)

Primary note proof

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Study-note section NET17

Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Question sources

Data Communication & Networking practice questions · Question 12

Question 109 | Q01533

A new network backbone needs high capacity, long reach and immunity to electromagnetic interference in the transmission medium. Which listed medium carries light rather than electrical signals and best matches these requirements?

A. coaxial cable

B. twisted pair cable

C. optical fiber

D. electrical cable

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer C: optical fiber

Optical fiber guides light, resists electromagnetic interference and supports high-capacity, long-reach links with suitable optics and installation. Twisted pair and coaxial cable carry electrical signals; their shielding or pair geometry can reduce interference but does not give the medium optical immunity. Exact speeds depend on the selected specification and equipment, so this comparison does not rank every possible fiber and copper product by speed.

Why each choice is right or wrong

A - Incorrect: Coaxial cable uses electrical signaling despite its shielding.

B - Incorrect: Twisted pair uses electrical signaling and reduces, rather than eliminates, coupled interference.

C - Correct: Optical fiber carries light and matches the stated capacity, reach and EMI requirements.

D - Incorrect: An electrical cable does not carry the required optical signal.

Primary note proof

Study-note section NET07

> Twisted pairs reduce coupled interference; UTP has no added shield and STP adds shielding. Coax has an inner conductor and outer shield sharing an axis. Fiber guides light and resists electromagnetic interference; it supports high capacity and long reach, but requires suitable optics and installation. Exact speed, cost and reach depend on the specification, not the material alone. Fiber can still be tapped.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 49-54

> Wavelength-division multiplexing (WDM): is designed to use the high-data-rate capability of fiber-optic cable. The optical fiber data rate is higher than the data rate of metallic transmission cable. Using a fiber-optic cable for one single line wastes the available bandwidth. Multiplexing allows us to combine several lines into one.

Data Communication and Computer Networking module | PDF page 11 | lines 23-33

> Twisted-pair cable: Consists of two conductors (normally copper), each with its own plastic insulation, twisted together.
> Two types
> • Unshielded Twisted-Pair cable (UTP)
> • Shielded Twisted-Pair cable (STP)
> Coaxial cable:- uses copper wire to conduct the signals electronically
> Fiber-optic cable:- uses glass or plastic conductor and transmits the signals as light
> Wireless/Unguided media the signal propagates in the atmosphere and in outer space

Question sources

Data Communication & Networking practice questions · Question 13

Question 110 | Q01534

Bits can be sent over guided and unguided media as analog signal by _____

A. digital modulation

B. amplitude modulation

C. frequency modulation

D. phase modulation

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer A: digital modulation

Digital modulation explicitly converts digital data to analog signals; ASK/FSK/PSK are its given methods. The other names are listed as analog modulation of analog data. Evidence: PDF 13.

Why each choice is right or wrong

A - Correct: Exactly the digital-to-analog conversion category (PDF 13.)

B - Incorrect: AM appears as analog modulation, not the digital-data category (PDF 13.)

C - Incorrect: FM appears as analog modulation, while digital frequency encoding is FSK (PDF 13.)

D - Incorrect: PM appears as analog modulation, while digital phase encoding is PSK (PDF 13.)

Primary note proof

Study-note section NET15

ASK varies carrier amplitude, FSK frequency, PSK phase; QAM combines amplitude and phase. AM/FM/PM can carry analog data. PCM digitizes analog input by sampling, quantizing and encoding. Sampling must exceed twice the highest retained baseband frequency in practical designs; b-bit samples have 2^b quantization levels and uncompressed rate fs×b. PAM/PWM are pulse modulation methods; pulse amplitude or width alone is not a binary PCM code.

Study-note section NET04

Analog signals vary continuously; digital signals use discrete symbol states. Data and signal types are separate: digital bits can modulate an analog carrier. A sine wave has amplitude, frequency and phase. Frequency f counts cycles/second, period T=1/f, and wavelength λ=v/f depends on propagation speed v. Phase specifies displacement within a cycle. Composite signals combine frequencies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 13 | lines 65-77

• Digital modulation is used to convert digital data to
analog signal. Ex ASK, FSK, PSK.
Analog Modulation can be accomplished in three ways:
• Amplitude modulation (AM)
• Frequency modulation (FM)
• Phase modulation (PM).
Digital Modulation Types (Digital to Analog signal
conversion)
Digital modulation is used to convert digital data to analog
signal. It can be accomplished in the following ways:
• Amplitude Shift Keying(ASK)
• Frequency Shift keying (FSK)
• Phase Shift keying (PSK)

Question sources

Data Communication & Networking practice questions · Question 14

Question 111 | Q01535

The portion of physical layer that interfaces with the media access control sublayer is called _____

A. physical signalling sublayer

B. physical data sublayer

C. physical address sublayer

D. physical transport sublayer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: Physical Signaling (PLS) sublayer in legacy IEEE 802.3

NET09 gives the architecture-specific name Physical Signaling (PLS) for the PHY portion interfacing with MAC in legacy IEEE 802.3 AUI/MAU terminology. That directly matches A. The qualification matters because modern PHY descriptions can use other sublayers; this is not a universal OSI sublayer name. The other offered names do not match the defined legacy interface.

Why each choice is right or wrong

A - Correct: Physical Signaling, abbreviated PLS, is the MAC-facing sublayer identified for the legacy IEEE architecture.

B - Incorrect: Physical data sublayer is not the MAC-facing sublayer name defined in NET09.

C - Incorrect: Physical address concerns addressing terminology; it is not the stated legacy PHY interface sublayer.

D - Incorrect: Physical transport sublayer is not the architecture-specific PLS name established here.

Primary note proof

Study-note section NET09

> In the legacy IEEE 802.3 architecture with an attachment unit interface (AUI) and medium attachment unit (MAU), the MAC interfaces to the Physical Signaling (PLS) sublayer. For this legacy MAC interface, the relevant sublayer is PLS. This name is architecture-specific: other PHY designs use reconciliation (RS), physical coding (PCS), physical medium attachment (PMA) and related sublayers. Do not present PLS as a universal OSI sublayer name.

Study-note section NET09

> A NIC supplies a computer’s network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which

device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Question sources

Data Communication & Networking practice questions · Question 15

Question 112 | Q01536

Which option gives the most complete combined set of physical-layer specifications among those listed?

A. mechanical specifications of electrical connectors and cables

B. electrical specification of transmission line signal level

C. specification for IR over optical fiber

D. all of the mentioned

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: All are physical-medium/signaling specifications

The physical layer specifies how signals and physical interfaces carry bits. Mechanical connector and cable requirements, electrical signal levels and optical signaling parameters all concern that role. Infrared signaling over suitable optical fiber is an optical physical-medium specification. Each individual item is therefore a valid example, but only the aggregate includes all three and satisfies the requested most-complete selection.

Why each choice is right or wrong

A - Incorrect: Mechanical connectors and cables are physical-layer specifications, but this alone omits the other valid items.

B - Incorrect: Electrical signal levels are physical-layer specifications, but this alone omits the other valid items.

C - Incorrect: Infrared optical signaling is a physical-medium specification, but this alone omits the other valid items.

D - Correct: All three items concern physical interfaces or signaling, so this is the complete combined response.

Primary note proof

Study-note section NET09

A NIC supplies a computer's network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Study-note section NET-S07

Fiber-optic links can use near-infrared light. The selected optical wavelength is part of the link's physical signaling specification, just as connectors and electrical signal levels are physical-interface specifications. This does not require every fiber link to use the same wavelength.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 11 | lines 23-33

Twisted-pair cable: Consists of two conductors (normally copper), each with its own plastic insulation, twisted together.
Two types
• Unshielded Twisted-Pair cable (UTP)
• Shielded Twisted-Pair cable (STP)
Coaxial cable:- uses copper wire to conduct the signals electronically
Fiber-optic cable:- uses glass or plastic conductor and transmits the signals as light
Wireless/Unguided media the signal propagates in the atmosphere and in outer space

Question sources

Data Communication & Networking practice questions · Question 16

Question 113 | Q01537

Which single option gives the complete set of listed operations that can be implemented as physical-layer transmission coding or modulation?

A. line coding

B. channel coding

C. modulation

D. all of the mentioned

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: all of the mentioned

Physical transmission includes mapping bits to signal patterns and modulating carriers. Channel coding in this question means redundancy used in transmission coding, rather than every possible higher-layer error-control mechanism. Under that scope, all three operations can be physical-layer work. Each individual choice names only part of the qualifying set, so the inclusive option is the complete response.

Why each choice is right or wrong

A - Incorrect: Line coding is physical signaling work, but alone omits the other qualifying operations.

B - Incorrect: Physical channel coding adds transmission redundancy, but alone is incomplete.

C - Incorrect: Modulation is a physical transmission operation, but alone is incomplete.

D - Correct: Line coding, physical channel coding and modulation all qualify, making this the complete set.

Primary note proof

Study-note section NET09

> A NIC supplies a computer’s network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Study-note section NET15

> Line coding maps digital bits to digital signal patterns. NRZ uses levels without a forced mid-bit return; Manchester supplies a mid-bit transition for clocking, with bit convention specified. Differential Manchester encodes using transitions relative to previous state. AMI alternates the

polarity of successive ones; pseudoternary alternates zero pulses. Block coding adds code redundancy; scrambling replaces problematic patterns without necessarily increasing bit count. Multilevel schemes include 2B1Q and 8B6T; MLT-3 uses a sequence of voltage transitions.

Study-note section NET15

ASK varies carrier amplitude, FSK frequency, PSK phase; QAM combines amplitude and phase. AM/FM/PM can carry analog data. PCM digitizes analog input by sampling, quantizing and encoding. Sampling must exceed twice the highest retained baseband frequency in practical designs; b-bit samples have 2^b quantization levels and uncompressed rate fs×b. PAM/PWM are pulse modulation methods; pulse amplitude or width alone is not a binary PCM code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 13 | lines 42-50

Digital Data to Digital Signal conversion
Coding methods are used to convert digital data into digital signals. There are two types of coding methods:
• Line Coding
• Block Coding
Scrambling is also one of the ways to convert digital data to digital signals but is not used.
Line Coding: It is the process of converting Digital data into digital signal.

Data Communication and Computer Networking module | PDF page 13 | lines 65-77

• Digital modulation is used to convert digital data to analog signal. Ex ASK, FSK, PSK.
Analog Modulation can be accomplished in three ways:
• Amplitude modulation (AM)
• Frequency modulation (FM)
• Phase modulation (PM).
Digital Modulation Types (Digital to Analog signal conversion)
Digital modulation is used to convert digital data to analog signal. It can be accomplished in the following ways:
• Amplitude Shift Keying(ASK)
• Frequency Shift keying (FSK)
• Phase Shift keying (PSK)

Data Communication and Computer Networking module | PDF page 13 | lines 27-37

Redundancy
In order to detect and correct the errors in the data communication we add some extra bits to the original data.
These extra bits are nothing but the redundant bits which will be removed by the receiver after receiving the data.
There are different techniques used for transmission error detection and correction.
Detection methods:
• Parity Check
• Cyclic Redundancy Check
• Checksum

Question sources

Data Communication & Networking practice questions · Question 17

Question 114 | Q01538

The physical layer translates logical communication requests from the _____ into hardware specific operations.

A. data link layer

B. network layer

C. trasnport layer

D. application layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: data link layer

The module adjacent layer order places data link immediately above physical, and data link passes frames to physical for bit transmission. Evidence: PDF 14, 14, 15.

Why each choice is right or wrong

A - Correct: Data link directly supplies the physical layer (PDF 14, 14, 15.)

B - Incorrect: Network supplies data link first, not physical directly (PDF 14, 14, 15.)

C - Incorrect: Transport supplies network first (PDF 14, 14, 15.)

D - Incorrect: Application passes through several intervening layers (PDF 14, 14, 15.)

Primary note proof

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Study-note section NET09

> A NIC supplies a computer's network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 12-29

> The Open Systems Interconnection (OSI) Model was
> developed by International Organization for Standardization
> (ISO). ISO is the organization, OSI is the model
> It was developed to allow systems with different platforms
> (Hardware & software) to communicate with each other.
> Now, it is a network model that defines the protocols for
> network communications.
> It is a hierarchical model that groups its processes into seven
> (7) layers. (Top to Bottom)
> 1. Application Layer
> 2. Presentation Layer
> 3. Session Layer
> 4. Transport Layer
> 5. Network Layer
> 6. Data Link Layer
> 7. Physical Layer

Each layer has specific functions to perform and has to co-operate with the layers above and below it.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes

access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Question sources

Data Communication & Networking practice questions · Question 18

Question 115 | Q01539

A single channel is shared by multiple signals by _____

A. analog modulation

B. digital modulation

C. multiplexing

D. phase modulation

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer C: multiplexing

The module defines multiplexing as simultaneous multiple signals across a single data link. Evidence: PDF 20, 13.

Why each choice is right or wrong

A - Incorrect: Analog modulation changes a carrier rather than defining shared-channel combination (PDF 20, 13.)

B - Incorrect: Digital modulation converts digital data to analog signals (PDF 20, 13.)

C - Correct: Multiplexing is the explicit shared-channel technique (PDF 20, 13.)

D - Incorrect: Phase modulation is one carrier parameter method (PDF 20, 13.)

Primary note proof

Study-note section NET32

Multiplexing combines multiple input streams onto a shared link; demultiplexing separates them at the receiver. It differs from broadcasting one message to many destinations. FDM assigns distinct frequency bands with suitable guard spacing; WDM assigns different optical wavelengths. TDM allocates time slots: synchronous TDM reserves recurrent slots even if an input is idle, while statistical TDM allocates according to demand and needs identification.

Study-note section NET15

ASK varies carrier amplitude, FSK frequency, PSK phase; QAM combines amplitude and phase. AM/FM/PM can carry analog data. PCM digitizes analog input by sampling, quantizing and encoding. Sampling must exceed twice the highest retained baseband frequency in practical designs; b-bit samples have 2^b quantization levels and uncompressed rate fs×b. PAM/PWM are pulse modulation methods; pulse amplitude or width alone is not a binary PCM code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 37-42

MULTIPLEXING
Whenever the bandwidth of a medium linking two devices is greater than the bandwidth needs of the devices, the link can be shared. Multiplexing is the set of techniques that allows the simultaneous transmission of multiple signals across a single data link.

Data Communication and Computer Networking module | PDF page 13 | lines 65-77

• Digital modulation is used to convert digital data to analog signal. Ex ASK, FSK, PSK.
Analog Modulation can be accomplished in three ways:
• Amplitude modulation (AM)
• Frequency modulation (FM)
• Phase modulation (PM).
Digital Modulation Types (Digital to Analog signal conversion)
Digital modulation is used to convert digital data to analog signal. It can be accomplished in the following ways:
• Amplitude Shift Keying(ASK)
• Frequency Shift keying (FSK)
• Phase Shift keying (PSK)

Question sources

Data Communication & Networking practice questions · Question 19

Question 116 | Q01540

The data link layer takes the packets from _____ and encapsulates them into frames for transmission.

A. network layer

B. physical layer

C. transport layer

D. application layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: network layer

The sending-side data link explicitly receives network-layer data and frames it for physical transmission. Evidence: PDF 15, 16, 15.

Why each choice is right or wrong

A - Correct: Network is the immediately preceding sender-side layer (PDF 15, 16, 15.)

B - Incorrect: Physical is the lower receive-side source of bits, not upper packets (PDF 15, 16, 15.)

C - Incorrect: Transport provides segments to network first (PDF 15, 16, 15.)

D - Incorrect: Application is several layers above framing (PDF 15, 16, 15.)

Primary note proof

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

> Data Link Layer
> • The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
> • Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
> • Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
> • Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.

• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver

(Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Question sources

Data Communication & Networking practice questions · Question 20

Question 117 | Q01541

Wireless transmission of signals can be done via _____

A. radio waves

B. microwaves

C. infrared

D. all of the mentioned

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer D: all of the mentioned

The module explicitly lists radio waves, microwaves and infrared as three unguided wireless media. Evidence: PDF 10.

Why each choice is right or wrong

A - Incorrect: Radio waves are listed (PDF 10.)

B - Incorrect: Microwaves are listed (PDF 10.)

C - Incorrect: Infrared is listed (PDF 10.)

D - Correct: All three appear in the unguided-media list (PDF 10.)

Primary note proof

Study-note section NET07

Radio, microwave and infrared are unguided media. Ground waves follow Earth, sky waves can reflect from ionized layers, and line-of-sight paths require suitable visibility. Microwaves are radio waves in the broader physical classification. Infrared generally does not penetrate walls. Directionality depends on antennas: omnidirectional means broad azimuth coverage, not equal radiation in every 3-D direction. Legacy Ethernet straight-through cables join unlike MDI/MDI-X roles; crossover joins like roles, such as PC-PC or switch-switch. Auto-MDI-X can remove that requirement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 10 | lines 36-55

TRANSMISSION MEDIA
Transmission media is a means by which a communicationsignal is carried from one system to another
A transmission medium can be defined as anything that cancarry information from a source to a destination. The transmission medium is usually free space, metallic cable orfiber optic cable. Guided and Unguided (WIRELESS) Guided Transmission Media uses a cabling system that guidesthe data signals along a specific path. Out of these twisted-paircable, coaxial cable transport signals in the form of electricsignals and fiber-optic cable transport signals in the form oflight. Unguided Transmission Media transport data without using a physical conductor. It uses wireless electromagnetic signals to send data. Three types of unguided media:
• Radio waves
• Micro waves
• Infrared.

Question sources

Data Communication & Networking practice questions · Question 21

Question 118 | Q01542

Under the OSI role convention used here, which task belongs to physical transmission coding rather than data-link framing and control? Here channel coding means physical-channel coding, not a link-frame CRC or retransmission procedure.

A. framing

B. error control

C. flow control

D. channel coding

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: channel coding

The data-link layer packages data into frames and can provide hop-local error and flow control. Physical transmission coding concerns how the channel carries the signal and any coding assigned to that physical role. The explicit convention distinguishes physical-channel coding from link error-detection redundancy or ARQ; without that scope, the broad term coding could overlap link functions.

Why each choice is right or wrong

A - Incorrect: Framing packages hop-delivery data into link frames.

B - Incorrect: Link error control includes detection and protocol-dependent recovery.

C - Incorrect: Link flow control regulates hop-local sending when supported.

D - Correct: Physical-channel coding is outside the stated link framing/control role.

Primary note proof

Study-note section NET13

> Link-layer LLC identifies or multiplexes upper-layer protocols and may supply link control; MAC handles medium-dependent addressing/access and frame delivery. Network addressing belongs to layer 3. Ethernet, PPP and HDLC are link protocols. Frame headers may carry addresses and control/type fields; synchronization and exact framing depend on the protocol.

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Study-note section NET09

> A NIC supplies a computer’s network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

> Data Link Layer
> • The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
> • Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits

into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 13 | lines 42-50

Digital Data to Digital Signal conversion
Coding methods are used to convert digital data into digital signals. There are two types of coding methods:
• Line Coding
• Block Coding
Scrambling is also one of the ways to convert digital data to digital signals but is not used.
Line Coding: It is the process of converting Digital data into digital signal.

Data Communication and Computer Networking module | PDF page 13 | lines 27-37

Redundancy
In order to detect and correct the errors in the data communication we add some extra bits to the original data. These extra bits are nothing but the redundant bits which will be removed by the receiver after receiving the data.
There are different techniques used for transmission error detection and correction.
Detection methods:
• Parity Check
• Cyclic Redundancy Check
• Checksum

Question sources

Data Communication & Networking practice questions · Question 22

Question 119 | Q01543

Which sublayer of the data link layer performs data link functions that depend upon the type of medium?

A. logical link control sublayer

B. media access control sublayer

C. network interface control sublayer

D. error control sublayer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Media Access Control (MAC)

The decisive phrase is “depend upon the type of medium.” NET13 assigns medium-dependent addressing, access and frame delivery to MAC. LLC instead identifies or multiplexes upper-layer protocols. The other names do not designate the LLC/MAC subdivision established in the note, so they cannot replace the specific MAC role.

Why each choice is right or wrong

A - Incorrect: LLC identifies upper-layer protocols and supplies link control rather than the medium-dependent access role.

B - Correct: MAC performs the medium-dependent addressing and access functions explicitly named in NET13.

C - Incorrect: A network interface is hardware/functionality, not the named medium-dependent data-link sublayer in this division.

D - Incorrect: Error control is a function; it is not the LLC/MAC sublayer name for medium-dependent access.

Primary note proof

Study-note section NET13

> Link-layer LLC identifies or multiplexes upper-layer protocols and may supply link control; MAC handles medium-dependent addressing/access and frame delivery. Network addressing belongs to layer 3. Ethernet, PPP and HDLC are link protocols. Frame headers may carry addresses and control/type fields; synchronization and exact framing depend on the protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 21 | lines 6-26

MEDIUM ACCESS CONTROL
The problem of controlling the access to the medium is similarto the rules of speaking in a meeting (the right to speak isupheld; two people do not speak at the same time; do notinterrupt each other; do not monopolize the discussion; ...) Three categories of multiple access protocols
Random Access Protocols - try your best like taxis do
• MA - Multiple Access
• CSMA - Carrier Sense MA
• CSMA/CD - CSMA with Collision Detection
• CSMA/CA - CSMA with Collision AvoidanceControlled-Access Protocols - get permission
• Reservation
• Polling

• Token Passing
Channelization Protocols - simultaneous use
• FDMA - Frequency-Division MA
• TDMA - Time-Division MA
• CDMA - Code-Division MA

Question sources

Data Communication & Networking practice questions · Question 23

Question 120 | Q01544

Which listed field identifies source or destination endpoints in a link-frame header, rather than synchronization or frame/control identification? Select one.

A. synchronization bytes

B. addresses

C. frame identifier

D. all of the mentioned

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B: addresses

Address fields identify the source or destination endpoints associated with a frame. Synchronization information helps delimit or align reception, while a frame identifier or control field distinguishes a frame or its function. The question asks specifically for endpoint identification, so the address option is the complete answer. Exact field placement depends on the link protocol and is not assumed universal.

Why each choice is right or wrong

A - Incorrect: Synchronization concerns alignment or boundaries rather than endpoint identity.

B - Correct: Address fields identify frame endpoints, matching the requested role.

C - Incorrect: A frame identifier distinguishes a frame or control function rather than its endpoint addresses.

D - Incorrect: The other listed fields have different roles, so all is not the endpoint-identification answer.

Primary note proof

Study-note section NET13

Link-layer LLC identifies or multiplexes upper-layer protocols and may supply link control; MAC handles medium-dependent addressing/access and frame delivery. Network addressing belongs

to layer 3. Ethernet, PPP and HDLC are link protocols. Frame headers may carry addresses and control/type fields; synchronization and exact framing depend on the protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 26 | lines 14-28

medium. TCP/IP was designed to be independent of the network access method, frame format, and medium. In this way, TCP/IP can be used to connect differing network types. These include LAN technologies such as Ethernet and Token Ring and WAN technologies such as X.25 and Frame Relay. Independence from any specific network technology gives TCP/IP the ability to be adapted to new technologies such as Asynchronous Transfer Mode (ATM).

Network Access layer uses a physical address to identify hosts and to deliver data.
The Network Access layer PDU is called a frame. It contains the IP packet as well as a protocol header and trailer from this layer.

Question sources

Data Communication & Networking practice questions · Question 24

Question 121 | Q01545

CRC stands for _____

A. cyclic redundancy check

B. code repeat check

C. code redundancy check

D. cyclic repeat check

Hint

Find the full expansion in the terminology note and compare each word rather than only matching initials.

Answer A: cyclic redundancy check

The error-detection list explicitly spells out Cyclic Redundancy Check, matching CRC initials. Evidence: PDF 13.

Why each choice is right or wrong

A - Correct: Cyclic Redundancy Check is the exact listed phrase (PDF 13.)

B - Incorrect: Code repeat check is not the expansion listed (PDF 13.)

C - Incorrect: Code redundancy check changes Cyclic to Code (PDF 13.)

D - Incorrect: Cyclic repeat check changes Redundancy to repeat (PDF 13.)

Primary note proof

Study-note section NET-S04

CRC expands to Cyclic Redundancy Check.

Study-note section NET13

Detection adds redundancy but does not guarantee correction. Even parity adds a bit so the total number of ones is even: 1011 needs parity 1. It detects every odd number of bit flips but misses even counts. CRC treats bits as a polynomial and appends the XOR-division remainder; the receiver repeats the division. Internet checksums add words with end-around carry and complement the sum. A burst spans from the first corrupted bit through the last, including

unchanged interior bits. Retransmission or sufficient forward-error-correcting redundancy is needed for recovery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 13 | lines 27-37

Redundancy
In order to detect and correct the errors in the data communication we add some extra bits to the original data.
These extra bits are nothing but the redundant bits which will be removed by the receiver after receiving the data.
There are different techniques used for transmission error detection and correction.
Detection methods:
• Parity Check
• Cyclic Redundancy Check
• Checksum

Question sources

Data Communication & Networking practice questions · Question 25

Question 122 | Q01546

Which single option gives the complete set of listed data-link protocols?

A. ethernet

B. point to point protocol

C. hdlc

D. all of the mentioned

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer D: all of the mentioned

Ethernet, PPP and HDLC are all link protocols. The question asks for the complete set among the listed choices, so selecting any one protocol alone omits two other valid members. The inclusive response therefore supplies the complete answer. This classification concerns the data-link protocol role and does not deny that a technology such as Ethernet also specifies physical signaling.

Why each choice is right or wrong

A - Incorrect: ethernet is a link protocol, but this option alone omits the other qualifying protocols.

B - Incorrect: point to point protocol is a link protocol, but this option alone omits the other qualifying protocols.

C - Incorrect: hdlc is a link protocol, but this option alone omits the other qualifying protocols.

D - Correct: Ethernet, PPP and HDLC all qualify; the combined option gives the complete set.

Primary note proof

Study-note section NET13

> Link-layer LLC identifies or multiplexes upper-layer protocols and may supply link control; MAC handles medium-dependent addressing/access and frame delivery. Network addressing belongs to layer 3. Ethernet, PPP and HDLC are link protocols. Frame headers may carry addresses and control/type fields; synchronization and exact framing depend on the protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 26 | lines 14-28

> medium. TCP/IP was designed to be independent
> of the network access method, frame format, and
> medium. In this way, TCP/IP can be used to
> connect differing network types. These include
> LAN technologies such as Ethernet and Token
> Ring and WAN technologies such as X.25 and
> Frame Relay. Independence from any specific
> network technology gives TCP/IP the ability to
> be adapted to new technologies such as
> Asynchronous Transfer Mode (ATM).
> Network Access layer uses a physical address to
> identify hosts and to deliver data.
> The Network Access layer PDU is called a frame.
> It contains the IP packet as well as a protocol
> header and trailer from this layer.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

> Data Link Layer
> • The Data Link layer adds reliability to the physical
> layer by providing error detection and correction
> mechanisms.
> • Framing: the Data Link layer receives the data
> from Network Layer and divides the stream of bits
> into fixed size manageable units called as Frames
> and sends it to the physical layer. On the receiver
> side, the data link layer receives the stream of bits
> from the physical layer and regroups them into
> frames and sends them to the Network layer.
> • Physical Addressing: The Data link layer appends
> the physical address in the header of the frame

before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Question sources

Data Communication & Networking practice questions · Question 26

Question 123 | Q01547

Which option gives the complete set of listed CSMA multiple-access mechanisms for shared-channel contention?

A. CSMA/CD

B. CSMA/CA

C. Both CSMA/CD & CSMA/CA

D. HDLC

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer C: Both CSMA/CD & CSMA/CA

CSMA/CD and CSMA/CA both govern shared-medium access. Legacy half-duplex Ethernet uses collision detection and backoff, while Wi-Fi uses collision avoidance procedures with waiting, backoff and acknowledgments. Selecting only one omits another valid listed mechanism. HDLC is a link framing/control protocol rather than either of these CSMA contention mechanisms, so the combined CSMA option is complete.

Why each choice is right or wrong

A - Incorrect: CSMA/CD is a valid mechanism but omits the other listed CSMA mechanism.

B - Incorrect: CSMA/CA is a valid mechanism but omits the other listed CSMA mechanism.

C - Correct: This includes both listed CSMA contention mechanisms.

D - Incorrect: HDLC is a link framing/control protocol, not a listed CSMA medium-contention mechanism.

Primary note proof

Study-note section NET14

> Random access allows contention: ALOHA sends opportunistically; CSMA senses before sending. In legacy half-duplex Ethernet, CSMA/CD detects collision, sends a jam indication, stops, and retries after random backoff. The jam signals a collision, not its clearance. Wi-Fi 802.11 uses CSMA/CA with waiting/backoff and acknowledgments; sensing does not eliminate hidden-terminal collisions.

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Study-note section NET13

> Link-layer LLC identifies or multiplexes upper-layer protocols and may supply link control; MAC handles medium-dependent addressing/access and frame delivery. Network addressing belongs to layer 3. Ethernet, PPP and HDLC are link protocols. Frame headers may carry addresses and control/type fields; synchronization and exact framing depend on the protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 21 | lines 6-26

> MEDIUM ACCESS CONTROL
> The problem of controlling the access to the medium is similarto the rules of speaking in a meeting (the right to speak isupheld; two people do not speak at the same time; do notinterrupt each other; do not monopolize the discussion; ...) Three categories of multiple access protocols
> Random Access Protocols - try your best like taxis do
> • MA - Multiple Access
> • CSMA - Carrier Sense MA
> • CSMA/CD - CSMA with Collision Detection
> • CSMA/CA - CSMA with Collision AvoidanceControlled-Access Protocols - get permission
> • Reservation
> • Polling

• Token Passing
Channelization Protocols - simultaneous use
• FDMA - Frequency-Division MA
• TDMA - Time-Division MA
• CDMA - Code-Division MA

Question sources

Data Communication & Networking practice questions · Question 27

Question 124 | Q01548

The network layer is concerned with _____ of dat
a.

A. bits

B. frames

C. packets

D. bytes

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: packets

Despite the broken dat/a line ending, the identifiable concept is the network-layer unit: packets. Evidence: PDF 15, 15, 16, 14.

Why each choice is right or wrong

A - Incorrect: Bits are physical units (PDF 15, 15, 16, 14.)

B - Incorrect: Frames are data-link units (PDF 15, 15, 16, 14.)

C - Correct: Packets are explicitly formed and transmitted by network layer (PDF 15, 15, 16, 14.)

D - Incorrect: Bytes are not the named network PDU (PDF 15, 15, 16, 14.)

Primary note proof

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged

frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Question sources

Data Communication & Networking practice questions · Question 28

Question 125 | Q01549

For IPv4 as described in the note, which listed function is not provided by IP itself? Here “error control” means header checksum error detection.

A. routing

B. inter-networking

C. congestion control

D. error control

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer C: congestion control

IPv4 supports logical packet delivery across networks, including routing, and checks its header with a checksum. The use of error control therefore refers to detection, not reliable end-to-end recovery. TCP supplies congestion control to protect network capacity; IPv4 itself does not provide TCP’s congestion-control service. The question separates those protocol services instead of denying that the network layer ever detects errors.

Why each choice is right or wrong

A - Incorrect: Routing is explicitly a network-layer function, so it cannot be the requested exception.

B - Incorrect: Forwarding packets between networks is an established network-layer responsibility.

C - Correct: Congestion control of the kind protecting network capacity is a TCP service rather than a service supplied by IPv4 itself.

D - Incorrect: IPv4 detects header corruption with its checksum, so error detection is present under the stated meaning.

Primary note proof

Study-note section NET16

> The IPv4 checksum protects its header; IP does not lack error checking altogether. Identification, flags and offset support fragmentation. IPv4 sources and routers may fragment when allowed; reassembly occurs at the destination. If DF is set and the next MTU is too small, forwarding fails and an ICMP diagnostic may be returned.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET30

Routing computes paths; forwarding chooses a next hop for an individual packet. Select the matching route with the longest prefix, then apply the routing system's tie-breaking criteria. A default /0 matches when nothing more specific does. Static routes are configured; dynamic protocols exchange reachability. Distance vector uses neighbor estimates, while link state distributes link information and computes shortest paths. Interior protocols operate within an autonomous system; BGP exchanges routes between systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 27 | lines 5-31

Internet Control Message Protocol (ICMP):
works at the Network layer and is used by IP for many different services. ICMP is a management protocol and messagingservice provider for IP. The following are some common events and messages that ICMP relates to:

Destination Unreachable: If a router can‘t send an IP datagram any further, it uses ICMP to send a message back to the sender, advising it of the situation.
Buffer Full: If a router‘s memory buffer for receiving incoming datagrams is full, it will use ICMP to send out this message until the congestion abates.
Hops: Each IP datagram is allotted a certain number of routers, called hops, to pass through. If it reaches its limitof hops before arriving at its destination, the last router to receive that datagram deletes it. The executioner router then uses ICMP to send an obituary message, informing the sending machine of the demise of its datagram.
Ping (Packet Internet Groper) uses ICMP echo messages to check the physical and logical connectivity of machineson a network.
Traceroute Using ICMP timeouts, Traceroute is used to discover the path a packet takes as it traverses an Internetwork.

Question sources

Data Communication & Networking practice questions · Question 29

Question 126 | Q01550

A 4 byte IP address consists of _____

A. only network address

B. only host address

C. network address & host address

D. network address & MAC address

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer C: network address & host address

The module says an IP address encodes network and host numbers; physical/MAC addresses are separate. Evidence: PDF 18, 19, 17.

Why each choice is right or wrong

A - Incorrect: Only network omits the stated host part (PDF 18, 19, 17.)

B - Incorrect: Only host omits the stated network part (PDF 18, 19, 17.)

C - Correct: Network plus host matches the stated composition (PDF 18, 19, 17.)

D - Incorrect: MAC is a separate physical address, not part of the IP field (PDF 18, 19, 17.)

Primary note proof

Study-note section NET-S06

An IP address identifies a network interface rather than limiting a whole computer to one address. A computer or router can have several interfaces and addresses. Under a supplied IPv4 prefix, leading bits identify the network and the remaining bits identify an address within that network; a MAC address is a separate link-layer identifier.

Study-note section NET22

Write IPv4 as four decimal octets 0-255, or as 32 binary bits. Use weights 128,64,32,16,8,4,2,1 within each octet. Thus 11000001 10000011 00011011 11111111 is 193.131.27.255. A value such as 257 is impossible. Syntactic validity differs from assignability: an address ending .0 or .255 can be a host under some prefixes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP
addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.

255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to resolve thephysical address of a device on a network, where its logical address is known.
Physical address is the 48 bit address that is imprinted on theNIC or LAN card, Logical address is the Internet Address orcommonly known as IP address that is used to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is used bya device on the network to find its Internet address when it knows its physical address.

Question sources

Data Communication & Networking practice questions · Question 30

Question 127 | Q01551

Which single option gives the complete set of listed algorithms or routing approaches usable in network-layer route design?

A. shortest path algorithm

B. distance vector routing

C. link state routing

D. all of the mentioned

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: all of the mentioned

Distance-vector routing uses estimates received from neighboring routers. Link-state routing distributes link information and uses shortest-path computation to build routes. Therefore each named approach is usable in network-layer routing design. Because the question asks for the complete set, choosing any one named approach would omit other valid members; the combined response is the complete selection.

Why each choice is right or wrong

A - Incorrect: Shortest-path computation is usable, but this choice alone omits the other listed routing approaches.

B - Incorrect: Distance-vector routing is usable, but alone is an incomplete set.

C - Incorrect: Link-state routing is usable, but alone is an incomplete set.

D - Correct: All three approaches can be used, so the inclusive option gives the complete set.

Primary note proof

Study-note section NET30

> Routing computes paths; forwarding chooses a next hop for an individual packet. Select the matching route with the longest prefix, then apply the routing system's tie-breaking criteria. A default /0 matches when nothing more specific does. Static routes are configured; dynamic protocols exchange reachability. Distance vector uses neighbor estimates, while link state distributes link information and computes shortest paths. Interior protocols operate within an autonomous system; BGP exchanges routes between systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 52-64

> ROUTING AND ROUTING PROTOCOLS
> Routing is the act of moving information across an internetwork from a source to a destination. At the heart of such protocol is the routing algorithm that determines the path for a packet.
> • Routing requires a host or a router to have a routingtable which is constructed by the routing algorithm.
> • There are wo kinds of routing algorithms (routing tables): nonadaptive (static) and adaptive (dynamic)
> • And two kinds of protocols
> o Interior routing protocols
> o Exterior routing protocols

Data Communication and Computer Networking module | PDF page 48 | lines 10-21

> In general, routing involves the network topology, or the setup of hardware, that can effectively relay data. Standard protocols help to identify the best routes for data and to ensure quality transmission. Individual pieces of hardware such as routers are referred to as —nodes‖ in the network. Different algorithms and protocols can be used to figure out how to best route data packets, and which nodes should be used. For example, some data packets travel according to a distance vector model that primarily uses distance as a factor, whereas others use Link-State Protocol, which involves other aspects of a —best path‖ for data.

Question sources

Data Communication & Networking practice questions · Question 31

Question 128 | Q01552

In a multi-destination scheme that carries an explicit list of multiple selected targets in each packet, which statements are incorrect as general descriptions? Select all that apply.

A. is same as broadcast routing

B. contains the list of all destinations

C. data is not sent by packets

D. there are multiple receivers

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A, C: is same as broadcast routing; data is not sent by packets

A packet can carry a list identifying several selected destinations without addressing every destination in the broadcast scope. That makes universal equivalence with broadcasting false. The scheme is explicitly packet-based, so saying data are not sent by packets is also false. The list of targets and the presence of multiple receivers are true descriptions and therefore are not selected in this negative question.

Why each choice is right or wrong

A - Correct: Multi-destination delivery to listed targets is not necessarily broadcast to every destination in a scope.

B - Incorrect: A list of destinations is a supported multi-destination packet representation, making this statement correct.

C - Correct: NET30 explicitly describes multi-destination packets, contradicting the statement that data is not sent by packets.

D - Incorrect: Multiple targeted receivers are the defining destination-set property, so this statement is not incorrect.

Primary note proof

Study-note section NET30

> Flooding sends copies on eligible outgoing links with duplicate/lifetime controls; random walk picks a neighbor probabilistically. Unicast targets one destination, broadcast all in a scope, multicast a subscribed group and anycast one member sharing an address, normally nearest under routing metrics. Multi-destination packets can list multiple targets; this is not necessarily identical to broadcast. Group multicast trees use mechanisms such as reverse-path checks; Ethernet spanning tree is a different layer-2 mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 54-59

ICMP- Internet Control Message Protocol: It is a signaling mechanism used to inform the sender about datagram problemsthat occur during transit. It is used by intermediate devices.
IGMP- Internet Group Message Protocol: It is a mechanism that allows sending the same message to a group of recipients.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Question sources

Data Communication & Networking practice questions · Question 32

Question 129 | Q01553

The network layer protocol for internet is _____

A. ethernet

B. internet protocol

C. hypertext transfer protocol

D. file transfer protocol

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: internet protocol

IP is explicitly the network/Internetwork layer protocol. Evidence: PDF 17, 18, 26.

Why each choice is right or wrong

A - Incorrect: Ethernet is a network-access/LAN technology (PDF 17, 18, 26.)

B - Correct: Internet Protocol is the explicit network-layer protocol (PDF 17, 18, 26.)

C - Incorrect: HTTP is an application service in the port table (PDF 17, 18, 26.)

D - Incorrect: FTP is explicitly application file transfer (PDF 17, 18, 26.)

Primary note proof

Study-note section NET16

> IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Study-note section NET13

> Link-layer LLC identifies or multiplexes upper-layer protocols and may supply link control; MAC handles medium-dependent addressing/access and frame delivery. Network addressing belongs to layer 3. Ethernet, PPP and HDLC are link protocols. Frame headers may carry addresses and control/type fields; synchronization and exact framing depend on the protocol.

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 22-38

> Network Layer or IP
> Also called as the Internetwork Layer (IP). It holds the
> IP protocol which is a network layer protocol and is
> responsiblefor source to destination transmission of
> data.
> The Internetworking Protocol (IP) is a connection-less &
> unreliable protocol.

It is a best effort delivery service. i.e. there is no error checking in IP, it simply sends the data and relies on its underlying layers to get the data transmitted to the destination.Even if it is unreliable IP cannot be considered weak and useless; since it provides only the functionality that is requiredfor transmitting data thereby giving maximum efficiency.
Since there is no mechanism of error detection or correction in IP, there will be no delay introduced on a medium where thereis no error at all.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Data Communication and Computer Networking module | PDF page 26 | lines 14-28

medium. TCP/IP was designed to be independent of the network access method, frame format, and medium. In this way, TCP/IP can be used to connect differing network types. These include LAN technologies such as Ethernet and Token Ring and WAN technologies such as X.25 and Frame Relay. Independence from any specific network technology gives TCP/IP the ability to be adapted to new technologies such as Asynchronous Transfer Mode (ATM).
Network Access layer uses a physical address to identify hosts and to deliver data.
The Network Access layer PDU is called a frame. It contains the IP packet as well as a protocol header and trailer from this layer.

Question sources

Data Communication & Networking practice questions · Question 33

Question 130 | Q01554

ICMP is primarily used for _____

A. error and diagnostic functions

B. addressing

C. forwarding

D. routing

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: error and diagnostic functions

ICMP provides error reports, echo connectivity checks and timeout-based traceroute. These are error and diagnostic functions. Evidence: PDF 27, 17, 12.

Why each choice is right or wrong

A - Correct: Matches destination-unreachable, echo and timeout examples (PDF 27, 17, 12.)

B - Incorrect: ARP handles address resolution; ICMP is not the addressing scheme (PDF 27, 17, 12.)

C - Incorrect: Routers forward packets; ICMP reports conditions (PDF 27, 17, 12.)

D - Incorrect: A routing algorithm chooses paths; ICMP provides diagnostics rather than route choice (PDF 27, 17, 12.)

Primary note proof

Study-note section NET20

> ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target's MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Study-note section NET30

> Routing computes paths; forwarding chooses a next hop for an individual packet. Select the matching route with the longest prefix, then apply the routing system's tie-breaking criteria. A default /0 matches when nothing more specific does. Static routes are configured; dynamic protocols exchange reachability. Distance vector uses neighbor estimates, while link state distributes link information and computes shortest paths. Interior protocols operate within an autonomous system; BGP exchanges routes between systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 27 | lines 5-31

Internet Control Message Protocol (ICMP):
works at the Network layer and is used by IP for many different services. ICMP is a management protocol and messagingservice provider for IP. The following are some common events and messages that ICMP relates to:
Destination Unreachable: If a router can‘t send an IP datagram any further, it uses ICMP to send a message back to the sender, advising it of the situation.
Buffer Full: If a router‘s memory buffer for receiving incoming datagrams is full, it will use ICMP to send out this message until the congestion abates.
Hops: Each IP datagram is allotted a certain number of routers, called hops, to pass through. If it reaches its limitof hops before arriving at its destination, the last router to receive that datagram deletes it. The executioner router then uses ICMP to send an obituary message, informing the sending machine of the demise of its datagram.
Ping (Packet Internet Groper) uses ICMP echo messages to check the physical and logical connectivity of machineson a network.
Traceroute Using ICMP timeouts, Traceroute is used to discover the path a packet takes as it traverses an Internetwork.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to resolve thephysical address of a device on a network, where its logical address is known.
Physical address is the 48 bit address that is imprinted on theNIC or LAN card, Logical address is the Internet Address orcommonly known as IP address that is used to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is used bya device on the network to find its Internet address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device used tointerconnect different types of network that uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN

• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to forwardpackets.
• The list of routing tables provides directions to transfer the data to a particular network destination -various paths to forward the data.
• A router analyzes a destination IP address of a givenpacket and compares it with the routing table to decide the packet's next best path

Question sources

Data Communication & Networking practice questions · Question 34

Question 131 | Q01555

At a sending host, transport identifies data from different application endpoints using ports and forms segments or datagrams. To which OSI layer does it pass these units directly?

A. network layer

B. data link layer

C. application layer

D. physical layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: network layer

On the sending path, transport creates segments or datagrams and hands them to the network layer for packet delivery. The network layer then uses the data-link layer, which in turn uses physical signaling. Ports distinguish application endpoints; this multiplexing does not merge unrelated applications into one undifferentiated TCP byte stream. The answer follows the immediate downward interface in the OSI stack.

Why each choice is right or wrong

A - Correct: The network layer immediately receives transport units on the sending path.

B - Incorrect: Data link is below network, not immediately below transport.

C - Incorrect: Application is above transport rather than its downward delivery target.

D - Incorrect: Physical is below both network and data link.

Primary note proof

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Study-note section NET18

TCP setup creates a logical transport connection, not a reserved physical circuit. Ports allow multiplexing separate applications over IP; they do not combine all applications into one TCP byte stream. Do not assign TCP reliability to all layer-4 protocols merely because the reference layer lists reliability as a possible service.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.

• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Question sources

Data Communication & Networking practice questions · Question 35

Question 132 | Q01556

Which of the following are transport layer protocols used in networking?

A. TCP and FTP

B. UDP and HTTP

C. TCP and UDP

D. HTTP and FTP

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: TCP and UDP

TCP and UDP are explicitly transport protocols; FTP and HTTP are application services. Evidence: PDF 18, 18.

Why each choice is right or wrong

A - Incorrect: FTP makes this pair include an application protocol (PDF 18, 18.)

B - Incorrect: HTTP makes this pair include an application protocol (PDF 18, 18.)

C - Correct: Both are explicitly transport protocols (PDF 18, 18.)

D - Incorrect: Both are application services, not transport protocols (PDF 18, 18.)

Primary note proof

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

> TCP - Transmission Control Protocol: TCP is a reliable
> connection-oriented, reliable protocol. i.e. a connection is

established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Data Communication & Networking practice questions · Question 36

Question 133 | Q01557

User datagram protocol is called connectionless because _____

A. all UDP packets are treated independently by transport layer

B. it sends data as a stream of related packets

C. it is received in the same order as sent order

D. it sends data very quickly

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A: UDP treats datagrams independently without connection setup

Connectionless UDP sends independent datagrams without establishing a transport connection. NET17 explicitly separates this from TCP’s reliable ordered byte stream. Fast transmission may be desirable but is not the definition of connectionlessness, and UDP itself does not preserve send order or combine datagrams into an ordered related stream.

Why each choice is right or wrong

A - Correct: Independent datagram treatment without built-in setup directly explains why UDP is connectionless.

B - Incorrect: An ordered related byte stream describes TCP's service rather than UDP's independent datagrams.

C - Incorrect: UDP provides no built-in ordering guarantee, so reception need not match transmission order.

D - Incorrect: Speed is not the defining condition; connectionlessness concerns the absence of connection establishment and associated state.

Primary note proof

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

> TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
> UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Question sources

Data Communication & Networking practice questions · Question 37

Question 134 | Q01558

For one TCP connection, which single option gives the complete set of correct descriptions?

A. is a connection-oriented protocol

B. uses a three way handshake to establish a connection

C. receives data from application as a single stream

D. all of the mentioned

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer D: all of the mentioned

TCP establishes logical transport connection state through the SYN, SYN+ACK and ACK exchange. It then provides an ordered byte-stream service for that connection rather than preserving individual application message boundaries. All three descriptions apply to one TCP connection. Each individual response is therefore incomplete under the complete-set instruction, while the inclusive option collects all of the valid properties.

Why each choice is right or wrong

A - Incorrect: TCP is connection-oriented, but this choice alone omits its handshake and stream properties.

B - Incorrect: TCP uses a three-way handshake, but that alone is an incomplete set.

C - Incorrect: TCP accepts an application byte stream within a connection, but this alone is incomplete.

D - Correct: Connection orientation, three-way setup and a per-connection byte stream are all valid.

Primary note proof

Study-note section NET18

> TCP establishes state with SYN, SYN+ACK, then ACK. SYN consumes one sequence position even with no application bytes. An acknowledgment states the next expected sequence number. For client ISN=395 and server ISN=260: client sends SYN seq395; server sends SYN+ACK seq260 ack396; client sends ACK seq396 ack261. Later byte counts advance sequence numbers. FIN also consumes one sequence position.

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Question sources

Data Communication & Networking practice questions · Question 38

Question 135 | Q01559

A _____ is a TCP name for a transport service access point.

A. port

B. pipe

C. node

D. protocol

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A: port

Ports identify source/destination applications or processes for transport delivery, fitting a transport service access point in this question. Evidence: PDF 28, 16.

Why each choice is right or wrong

A - Correct: Port is the module application/process identifier (PDF 28, 16.)

B - Incorrect: Pipe is not the described TCP endpoint identifier (PDF 28, 16.)

C - Incorrect: Node is a network device rather than application endpoint (PDF 28, 16.)

D - Incorrect: Protocol names rules, not the numeric process endpoint (PDF 28, 16.)

Primary note proof

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 2-14

> TCP and UDP must use port numbers to communicate
> with the upper layers, because they‘re what keep track of
> different conversations crossing the network
> simultaneously. These port numbers identify the source
> and destination application or process in the TCP
> segment. There are 216 = 65,536 ports available.
> Well-known ports: The port numbers range from 0 to
> 1,023.
> Registered ports: The port numbers range from 1,024 to

49,151. Registered ports are used by applications or services that need to have consistent port assignments. Dynamic or private ports: The port numbers range from 49,152 to 65,535.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Question sources

Data Communication & Networking practice questions · Question 39

Question 136 | Q01560

Which phrase names the precise transport-layer endpoint-delivery responsibility in the OSI model?

A. application to application communication

B. process to process communication

C. node to node communication

D. man to man communication

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: process to process communication

Transport provides process-to-process delivery, with ports identifying transport endpoints. Application-to-application communication is a broader description of what that service supports, but it does not name the precise endpoint-delivery term requested here. Hop or node-to-node frame delivery belongs to the data-link layer. Human communication describes an application purpose rather than the protocol endpoint responsibility used to distinguish the OSI layers.

Why each choice is right or wrong

A - Incorrect: Applications use transport, but this broad description does not name the precise process-delivery term requested.

B - Correct: Process-to-process is the precise transport endpoint-delivery responsibility.

C - Incorrect: Hop/node-to-node frame delivery is a link-layer responsibility rather than transport process delivery.

D - Incorrect: Human-to-human interaction is not the protocol endpoint service defining transport.

Primary note proof

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

> • At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
> • To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
> • The data can be transported in a connection oriented or connectionless manner.

• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 28 | lines 2-14

TCP and UDP must use port numbers to communicate with the upper layers, because they‘re what keep track of different conversations crossing the network simultaneously. These port numbers identify the source and destination application or process in the TCP segment. There are 216 = 65,536 ports available.
Well-known ports: The port numbers range from 0 to 1,023.
Registered ports: The port numbers range from 1,024 to 49,151. Registered ports are used by applications or services that need to have consistent port assignments.
Dynamic or private ports: The port numbers range from 49,152 to 65,535.

Question sources

Data Communication & Networking practice questions · Question 40

Question 137 | Q01561

A local telephone network is an example of a _____ network.

A. Packet switched

B. Circuit switched

C. Bit switched

D. Line switched

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B: Circuit switched

Assuming the traditional local circuit telephone network, call setup creates a dedicated circuit and teardown releases it. This identifies circuit switching.

Why each choice is right or wrong

A - Incorrect: Packet switching is discrete packet communication

B - Correct: Circuit switching matches the traditional dedicated-call assumption

C - Incorrect: Bit switched is not a listed switching category

D - Incorrect: Line switched is not a listed switching category

Primary note proof

Study-note section NET31

Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 12-26

Circuit Switching: In circuit switching, two
communicating stations are connected by a dedicated
communication path which consists of intermediate
nodes in the network and the links that connect these
nodes.
It involved the following three distinct steps
• Circuit Establishment: To establish an end-to-end
connection before any transfer of data.
• Data transfer: Transfer data is from the source to the
destination. The data may be analog or digital,
depending on the nature of the network. The
connection is generally full-duplex.
• Circuit disconnect: Terminate connection at the end
of data transfer. Signals must be propagated to
deallocate the dedicated resources.

Data Communication and Computer Networking module | PDF page 20 | lines 27-36

Message Switching:
Packet Switching: Packet switching was designed to
address the shortcomings of circuit switching in dealing with
data communication. Unlike circuit switching where
communication is continuous along a dedicated circuit, in
packet switching, communication is discrete in form of packets.
The three switching modes to choose from are:
• Cut-through

- Store-and-forward
- Fragment-free

Question sources

Data Communication & Networking practice questions · Question 41

Question 138 | Q01562

Most packet switches use this principle _____

A. Stop and wait

B. Store and forward

C. Store and wait

D. Stop and forward

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer B: Store and forward

Packet store-and-forward means receiving the complete packet or frame before forwarding it. That is the switching principle named by B. Stop-and-wait is instead a reliability procedure that sends one frame and waits for acknowledgment. Cut-through alternatives exist, so the answer identifies the conventional principle without claiming every switch must use it.

Why each choice is right or wrong

A - Incorrect: Stop-and-wait is a link reliability/flow procedure involving acknowledgments, not the switching-buffering principle.

B - Correct: Store-and-forward buffers the full packet or frame before sending it on the next link.

C - Incorrect: Store and wait is not the defined switching principle named in NET31.

D - Incorrect: Stop and forward is not the defined receive-complete-then-forward switching procedure.

Primary note proof

Study-note section NET31

Datagram packets are forwarded independently. Virtual circuits establish forwarding state, but resource reservation is optional, not a necessary property of every virtual circuit. Message switching stores a complete message before forwarding, without a dedicated end-to-end path. Packet store-and-forward waits for the full packet/frame; cut-through starts after enough header arrives; fragment-free historically waits through the initial collision-sensitive portion. No routing algorithm guarantees that every packet always takes the geometrically shortest route.

Study-note section NET14

Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 27-36

Message Switching:
Packet Switching: Packet switching was designed to address the shortcomings of circuit switching in dealing with data communication. Unlike circuit switching where communication is continuous along a dedicated circuit, in packet switching, communication is discrete in form of packets.
The three switching modes to choose from are:
• Cut-through
• Store-and-forward
• Fragment-free

Question sources

Data Communication & Networking practice questions · Question 42

Question 139 | Q01563

Which is not a application layer protocol?

A. HTTP

B. SMTP

C. FTP

D. TCP

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: TCP

TCP is explicitly transport, whereas SMTP/FTP and the HTTP service belong to the application side. Evidence: PDF 18, 18, 28.

Why each choice is right or wrong

A - Incorrect: HTTP appears as an application service in the port table (PDF 18, 18, 28.)

B - Incorrect: SMTP is in the application protocol list (PDF 18, 18, 28.)

C - Incorrect: FTP is explicitly application file transfer (PDF 18, 18, 28.)

D - Correct: TCP is explicitly a transport protocol (PDF 18, 18, 28.)

Primary note proof

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the combined session, presentation, and application layers in the OSI model and many protocols are defined at this layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Data Communication & Networking practice questions · Question 43

Question 140 | Q01564

The packet of information at the application layer is called _____

A. Packet

B. Message

C. Segment

D. Frame

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Message

The module calls communicated information a message and transport receives upper-layer data before segmenting it. Message is the suitable upper/application unit among these options. Evidence: PDF 5, 16, 16, 15, 15.

Why each choice is right or wrong

A - Incorrect: Packet is network-layer terminology

B - Correct: Message names the communicated upper data

C - Incorrect: Segment is transport terminology

D - Incorrect: Frame is data-link terminology

Primary note proof

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 5 | lines 23-35

A Data Communication system has five components:-
• Message:- is the information to be communicated by the sender to the receiver.
• Sender The sender is any device that is capable of sending the data (message).
• Receiver: - is a device that the sender wants to communicate the data (message).
• Transmission Medium: - It is the path by which the message travels from sender to receiver. It can be wired or wireless and many subtypes in both.
• Protocol:- It is an agreed upon set or rules used by the sender and receiver to communicate data. Rules that govern data communication.

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.

• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize

devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Question sources

Data Communication & Networking practice questions · Question 44

Question 141 | Q01565

Which one of the following is an architecture paradigms?

A. Peer to peer

B. Client-server

C. HTTP

D. Both Peer-to-Peer & Client-Server

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: Both Peer-to-Peer & Client-Server

The architecture section explicitly describes client-server and peer-to-peer; HTTP is a protocol rather than that resource-service arrangement. Evidence: PDF 24, 18.

Why each choice is right or wrong

A - Incorrect: Peer-to-peer is an architecture, but the combined option includes the other valid architecture (PDF 24, 18.)

B - Incorrect: Client-server is an architecture, but the combined option includes the other valid architecture (PDF 24, 18.)

C - Incorrect: HTTP is a protocol/service, not either listed architecture paradigm (PDF 24, 18.)

D - Correct: Both named architectures are explicitly described (PDF 24, 18.)

Primary note proof

Study-note section NET02

An intranet serves an organization internally; an extranet exposes selected services to approved outsiders. Neither term alone specifies geographic size. A client requests a service; a server

supplies it. Peer systems may perform both roles. Dedicated servers concentrate on service workloads but can still be clients of DNS, backup or update services. A ten-user cutoff is a teaching guideline, not a technical law. Centralization simplifies administration but creates dependencies; peer networks still need security.

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 24 | lines 23-35

Network Architecture: based on the architecture (where do the clients get the shared resources?), networks can be categorized into three:
Client-Server Architecture: There can be one or more systems acting as Server. Other being Client, request the Server to serve requests. Servers take and process request on clients‘ behalf.
Peer-to-Peer (Point-to-point): Two systems can be connected Point-to-Point, or in other words back-to-back fashion. They both reside on same level and called peers.
There can be hybrid network which involves network architecture of both the above types.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Data Communication & Networking practice questions · Question 45

Question 142 | Q01566

Application developer has permission to decide the following on transport layer side

A. Transport layer protocol

B. Maximum buffer size

C. Both Transport layer protocol and Maximum buffer size

D. None of the mentioned

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: Choose transport protocol and request buffer size, subject to OS limits

NET17 permits an application to choose its transport protocol and request socket-buffer sizes. In that application-configuration sense, C combines both choices. “Maximum buffer size” must mean the requested socket limit, not authority to override the operating system’s enforced maximum; the note explicitly makes the request subject to OS limits.

Why each choice is right or wrong

A - Incorrect: Choosing a transport is permitted, but this option omits the also-permitted socket-buffer size request.

B - Incorrect: A buffer limit can be requested subject to OS limits, but this omits transport-protocol choice.

C - Correct: Both settings can be selected/requested by the application, with the buffer size constrained by the operating system.

D - Incorrect: At least transport choice and buffer-size requests are explicitly permitted, so none is false.

Primary note proof

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

> TCP - Transmission Control Protocol: TCP is a reliable
> connection-oriented, reliable protocol. i.e. a connection is
> established between the sender and receiver before the data can

be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 28 | lines 2-14

TCP and UDP must use port numbers to communicate with the upper layers, because they‘re what keep track of different conversations crossing the network simultaneously. These port numbers identify the source and destination application or process in the TCP segment. There are 216 = 65,536 ports available.
Well-known ports: The port numbers range from 0 to 1,023.
Registered ports: The port numbers range from 1,024 to 49,151. Registered ports are used by applications or services that need to have consistent port assignments.
Dynamic or private ports: The port numbers range from 49,152 to 65,535.

Question sources

Data Communication & Networking practice questions · Question 46

Question 143 | Q01567

Using “end-to-end” for application service and “process-to-process” for the transport service beneath it, which service does the application layer offer?

A. End to end

B. Process to process

C. Both End to end and Process to process

D. None of the mentioned

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: End to end

The question explicitly separates an application service from the underlying transport service. The note describes end-to-end application service as building on process-to-process transport. Under those named scopes, the application layer offers the end-to-end service, while process delivery identifies the transport boundary. Combining both would collapse the distinction being tested; none of the mentioned fails because the application-service term is present.

Why each choice is right or wrong

A - Correct: End-to-end is the stated application-service scope.

B - Incorrect: Process-to-process names the underlying transport service in this question.

C - Incorrect: The combination assigns the underlying transport-service term to the application layer, contrary to the stated distinction.

D - Incorrect: The stated application-service term is offered in A.

Primary note proof

Study-note section NET19

> The four-layer TCP/IP model folds OSI layers 5-7 into application, retains transport, calls layer 3 Internet, and combines link/physical as network access. A model is a guide to functions, not a guarantee that every application gets a reliable network. Distinguish end-to-end application service from its underlying process-to-process transport.

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

> Application Layer
> • The application layer enables the user to communicate
> its data to the receiver by providing certain services.
> • The main Responsibility of Application layer is to
> provide the user access to network resources.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

> • At the sending side, the transport layer receives data
> from the session layer, divides it into units called
> segments and sends it to the network layer.
> • To ensure process to process delivery the transport
> layer makes use of port address to identify the data

from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Question sources

Data Communication & Networking practice questions · Question 47

Question 144 | Q01568

To deliver a message to the correct application program running on a host, the _____ address must be consulte
d.

A. IP

B. MAC

C. Port

D. None of the mentioned

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: Port

Port numbers explicitly identify the destination application/process, while IP identifies an interface and MAC a local physical endpoint. Evidence: PDF 28, 18, 17.

Why each choice is right or wrong

A - Incorrect: IP identifies the host interface, not which program (PDF 28, 18, 17.)

B - Incorrect: MAC identifies a local physical interface, not which program (PDF 28, 18, 17.)

C - Correct: Port identifies the process/application (PDF 28, 18, 17.)

D - Incorrect: None fails because the port option is present (PDF 28, 18, 17.)

Primary note proof

Study-note section NET17

Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Study-note section NET-S06

An IP address identifies a network interface rather than limiting a whole computer to one address. A computer or router can have several interfaces and addresses. Under a supplied IPv4 prefix, leading bits identify the network and the remaining bits identify an address within that network; a MAC address is a separate link-layer identifier.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 2-14

TCP and UDP must use port numbers to communicate
with the upper layers, because they‘re what keep track of
different conversations crossing the network
simultaneously. These port numbers identify the source
and destination application or process in the TCP
segment. There are 216 = 65,536 ports available.
Well-known ports: The port numbers range from 0 to
1,023.
Registered ports: The port numbers range from 1,024 to
49,151. Registered ports are used by applications or
services that need to have consistent port assignments.
Dynamic or private ports: The port numbers range from
49,152 to 65,535.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to
resolve thephysical address of a device on a network,
where its logical address is known.
Physical address is the 48 bit address that is imprinted

on theNIC or LAN card, Logical address is the Internet Address orcommonly known as IP address that is used to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is used bya device on the network to find its Internet address when it knows its physical address.

Question sources

Data Communication & Networking practice questions · Question 48

Question 145 | Q01572

In TCP connection establishment procedure, if a client chooses Initial Sequence Number of 395, and server chooses 260, then:

A. The 1st segment sent will contain sequence number of 396

B. The 2nd segment sent will contain acknowledgement number of 261

C. The 3rd segment sent will contain sequence number of 261

D. The 2nd segment sent will contain acknowledgement number of 396

E. The 2nd segment sent will contain sequence number of 395

F. The 3rd segment sent will contain acknowledgement number of 396

Hint

Trace the handshake in order, keeping each endpoint’s sequence and acknowledgment fields separate.

Answer D: Second segment acknowledgment number 396

SYN consumes one sequence position. The first segment is client SYN seq395. The server’s second segment is SYN+ACK seq260 ack396, acknowledging the client SYN. The third is client ACK seq396 ack261. Comparing those fields shows only D matches; sequence numbers and acknowledgment numbers must not be swapped between endpoints.

Why each choice is right or wrong

A - Incorrect: The first client SYN uses its initial sequence number 395, not the next position 396.

B - Incorrect: The second segment acknowledges the client SYN with 396; 261 is the later acknowledgment of the server SYN.

C - Incorrect: The third segment’s sequence number is the client’s next position 396, not the server acknowledgment value 261.

D - Correct: The server’s SYN+ACK acknowledges client ISN 395 plus one, giving 396.

E - Incorrect: The second segment is sent by the server with its own sequence number 260.

F - Incorrect: The third segment acknowledges server ISN 260 plus one, giving 261 rather than 396.

Primary note proof

Study-note section NET18

TCP establishes state with SYN, SYN+ACK, then ACK. SYN consumes one sequence position even with no application bytes. An acknowledgment states the next expected sequence number. For client ISN=395 and server ISN=260: client sends SYN seq395; server sends SYN+ACK seq260 ack396; client sends ACK seq396 ack261. Later byte counts advance sequence numbers. FIN also consumes one sequence position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Question sources

Mock exam 3 · Question 4

Question 146 | Q01573

For IPv4 when fragmentation is permitted, which option gives the complete set of nodes that may fragment packets and the location of reassembly?

A. Fragment at the source host; re-assemble at the destination host

B. Fragment at the source host and routers; re-assemble at the destination host

C. Fragment at the source host and routers; re-assemble at the routers and destination host

D. Fragment at the source host; re-assemble at the routers and destination host

E. IP does not support Fragmentation and Re-assembly.

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer B: Fragment at the source host and routers; re-assemble at the destination host

IPv4 permits both the source and a router to fragment packets when fragmentation is allowed and an outgoing MTU requires it. The destination reassembles the fragments. Source-only fragmentation omits a permitted IPv4 case; it instead resembles the IPv6 restriction. Adding intermediate-router reassembly is incorrect. Naming IPv4 and asking for the complete permitted set makes the version-specific distinction explicit.

Why each choice is right or wrong

A - Incorrect: A source may fragment, but this option omits permitted IPv4 router fragmentation.

B - Correct: IPv4 sources and routers may fragment; only the destination reassembles.

C - Incorrect: Neither cited IP fragmentation rule places reassembly at intermediate routers.

D - Incorrect: Destination reassembly is required; the additional router reassembly claim is unsupported for either version.

E - Incorrect: Both version-specific rules support fragmentation and reassembly, although the allowed fragmenting nodes differ.

Primary note proof

Study-note section NET16

> The IPv4 checksum protects its header; IP does not lack error checking altogether. Identification, flags and offset support fragmentation. IPv4 sources and routers may fragment when allowed; reassembly occurs at the destination. If DF is set and the next MTU is too small, forwarding fails and an ICMP diagnostic may be returned.

Study-note section NET23

IPv6 has 128-bit hexadecimal addresses, flexible prefixes and no IPv4-style broadcast. Sources, not routers, fragment IPv6 packets. IPv6 does not automatically encrypt traffic or make authentication unnecessary. Compare IPv6's larger 128-bit address space and encryption support separately; specify the property being evaluated.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the Internet. Every host and router on the Internet has an IP address, which encodes its network number and host number.
• The combination is unique: in principle, no two machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination address fields of IP packets.
• An IP address does not refer to a host but it refers to a network interface.

Question sources

Mock exam 3 · Question 5

Question 147 | Q01574

Packets in one TCP connection can be distinguished from packets in another TCP connection by:

A. Source port, destination port, protocol number

B. Source IP address, destination IP address, protocol number

C. Source port, destination port, source IP address, destination IP address

D. Source IP address, destination IP address

E. Source port, destination port

F. Protocol number

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer C: Both IP addresses and both port numbers

All packets under discussion are already TCP, so their protocol identity is common. Distinguishing arbitrary simultaneous connections needs the endpoint four-tuple: source IP, destination IP, source port and destination port. Addresses alone cannot distinguish separate applications between the same hosts; ports alone can repeat between different hosts. C supplies all four required fields.

Why each choice is right or wrong

A - Incorrect: Ports and protocol can repeat between different host pairs, so this omits necessary IP addresses.

B - Incorrect: Two hosts can have several simultaneous TCP connections; addresses and TCP protocol alone do not distinguish them.

C - Correct: The source/destination IP addresses and source/destination ports form the TCP connection four-tuple.

D - Incorrect: An IP-address pair cannot distinguish multiple connections between the same two hosts.

E - Incorrect: The same source/destination port pair can be used with different IP-address endpoints.

F - Incorrect: Every connection in the stem uses TCP, so protocol number alone distinguishes none of them.

Primary note proof

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 2-14

TCP and UDP must use port numbers to communicate with the upper layers, because they‘re what keep track of different conversations crossing the network simultaneously. These port numbers identify the source and destination application or process in the TCP segment. There are 216 = 65,536 ports available.
Well-known ports: The port numbers range from 0 to 1,023.
Registered ports: The port numbers range from 1,024 to 49,151. Registered ports are used by applications or services that need to have consistent port assignments.
Dynamic or private ports: The port numbers range from 49,152 to 65,535.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the Internet. Every host and router on the Internet has an IP address, which encodes its network number and host number.
• The combination is unique: in principle, no two machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination address fields of IP packets.
• An IP address does not refer to a host but it refers to a network interface.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Question sources

Mock exam 3 · Question 6

Question 148 | Q01630

Compared to other common topologies in LANs (such as bus, mesh, ring), which of the following is a disadvantage of using a star topology for LAN?

A. Poor performance due to sharing of point-to-multipoint links between users

B. Failure of a single station results in no stations being able to communicate

C. Failure of a single link results in no stations being able to communicate

D. Each station requires many network interfaces

E. Too many links needed in the network

F. Failure of a special node results in no stations being able to communicate

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer F: Failure of a special node results in no stations being able to communicate

The star description explicitly calls its hub/switch a single point of failure: failure of that central device brings the whole system down. Ordinary station/link failure is isolated. Evidence: PDF 23, 23, 23, 23.

Why each choice is right or wrong

A - Incorrect: Star has separate central links, not the bus shared-backbone defect (PDF 23, 23, 23, 23.)

B - Incorrect: Failure of an ordinary station does not remove the central device (PDF 23, 23, 23, 23.)

C - Incorrect: The module says a link failure affects only its attached device (PDF 23, 23, 23, 23.)

D - Incorrect: Many interfaces per station is the full-mesh burden (PDF 23, 23, 23, 23.)

E - Incorrect: Star uses more cable than bus/ring but less than mesh, so this is not the distinctive universal comparison (PDF 23, 23, 23, 23.)

F - Correct: The special central node is explicitly a single point of failure (PDF 23, 23, 23, 23.)

Primary note proof

Study-note section NET08

> Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Study-note section NET08

A full undirected mesh of n nodes needs n(n-1)/2 links and n-1 interfaces per node. For 50 nodes: 50×49/2=1225 links. Redundant paths improve fault tolerance at greater cable and interface cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Data Communication and Computer Networking module | PDF page 23 | lines 45-53

Mesh Topology: in this topology devices are connected with many redundant interconnections between network nodes. In a full mesh topology, every node has a connection to every other node in the network, which makes it the most expensive topology over all the other topologies. The number of cables grows fast as the number of nodes increases, and it can be calculated by using the general formula ((n (n - 1)) /2), where n is the number of nodes in the

Data Communication and Computer Networking module | PDF page 23 | lines 4-18

Bus Topology: in this topology all devices are connected to a central cable, called the bus or backbone, which is terminated at its ends. The purpose of the terminators is to stop the signal from bouncing, thereby clearing the cable so that other computers can send data. Message transmitted along the Bus is visible to all computers connected to the backbone cable. As the message arrives at each workstation, the workstation checks the destination address contained in the message either to process or drop the packet if it matches or not respectively. Its

advantages are, ease of installation and less amount of cable requirement. Its main drawback is, the entire network will be shut down if there is a break in the main cable.

Data Communication and Computer Networking module | PDF page 23 | lines 35-44

Ring Topology: in this topology, all devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it. Some of its advantages include, easy to install and reconfigure, less expensive (than mesh), and performance is even despite the number of users. Its cons include, break in the ring (such as a disabled station) can disable the entire network, and limitations on media and traffic (limitation on ring length and number of devices).

Question sources

Mock exam 2 · Question 19

Question 149 | Q01631

For IPv4 networks, distinguish interface IP addresses, transport port numbers and the IPv4 Protocol header field. Which statement is true?

A. The protocol number commonly used by web server applications is 80

B. Every application must use a well-known port number

C. Each host and router in the Internet has only a single IP address

D. Client applications may dynamically select port numbers

E. A protocol number is used to indicate the version of IP being used

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer D: Client applications may dynamically select port numbers

Client applications commonly use dynamically selected ephemeral ports. Ports identify transport endpoints, whereas the IPv4 Protocol field identifies the carried upper-layer protocol, such as TCP or UDP. The separate Version field identifies the IP version. Port 80 is a usual HTTP service port, not an IP protocol number. Hosts and routers can have multiple addressed interfaces, so they are not restricted to one IP address.

Why each choice is right or wrong

A - Incorrect: 80 is a usual HTTP port number, not the IPv4 Protocol value for HTTP.

B - Incorrect: Clients may use ephemeral ports, so a well-known port is not mandatory for every application.

C - Incorrect: A device can have multiple addressed interfaces and therefore multiple IP addresses.

D - Correct: Clients may dynamically select ephemeral transport ports.

E - Incorrect: The IPv4 Protocol field identifies the upper-layer protocol; the Version field identifies the IP version.

Primary note proof

Study-note section NET-S06

An IP address identifies a network interface rather than limiting a whole computer to one address. A computer or router can have several interfaces and addresses. Under a supplied IPv4 prefix, leading bits identify the network and the remaining bits identify an address within that network; a MAC address is a separate link-layer identifier.

Study-note section NET-S06

TCP and UDP port fields are 16 bits, allowing numbers from 0 through 65535. Well-known ports are 0-1023, registered ports are 1024-49151, and dynamic/private ports are 49152-65535. Client endpoints can use dynamically selected ports instead of every application requiring a well-known assignment. These port numbers are distinct from the IP version and upper-protocol fields.

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 2-14

TCP and UDP must use port numbers to communicate
with the upper layers, because they‘re what keep track of
different conversations crossing the network
simultaneously. These port numbers identify the source
and destination application or process in the TCP
segment. There are 216 = 65,536 ports available.
Well-known ports: The port numbers range from 0 to
1,023.
Registered ports: The port numbers range from 1,024 to
49,151. Registered ports are used by applications or
services that need to have consistent port assignments.
Dynamic or private ports: The port numbers range from
49,152 to 65,535.

Data Communication and Computer Networking module | PDF page 28 | lines 23-31

Telnet 23 SNMP 161
SMTP 25 TFTP 69
HTTP 80 DNS 53
FTP 21 POP3 110
DNS 53 DHCP 68
HTTPS 443 NTP 123
SSH 22 RPC 530
Table. Sample TCP and UDP port numbers from well-known category

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the Internet. Every host and router on the Internet has an IP address, which encodes its network number and host number.
• The combination is unique: in principle, no two machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination address fields of IP packets.
• An IP address does not refer to a host but it refers to a network interface.

Question sources

Mock exam 2 · Question 20

Question 150 | Q01632

If you would like to give IP address to 57 hosts from your IP prefix which subnet mask do you use most effectively?

A. 255.255.255.248

B. 255.255.255.240

C. 255.255.255.224

D. 255.255.255.128

E. 255.255.255.192

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer E: 255.255.255.192

For 57 hosts choose the smallest offered block with at least 57 total addresses: /26 (192=11000000) has 64. The next smaller /27 has only 32, and /25 wastes twice the space. Even the conventional two reserved addresses leave /26 sufficient. Evidence: PDF 19, 19.

Why each choice is right or wrong

A - Incorrect: 248 gives only eight total addresses (PDF 19, 19.)

B - Incorrect: 240 gives only 16 (PDF 19, 19.)

C - Incorrect: 224 gives only 32 (PDF 19, 19.)

D - Incorrect: 128 gives 128 and uses more space than necessary (PDF 19, 19.)

E - Correct: 192 gives 64, the smallest offered sufficient block (PDF 19, 19.)

Primary note proof

Study-note section NET26

> For H hosts, choose the smallest h satisfying 2^h-2≥H, then p=32-h. For 24 hosts use /27; 52 or 57 use /26; 100 use /25; 500 use /23. For at least six equal subnets choose b=3. A Class C /24 split to /27 yields eight subnets. Always check both the number of subnets and the host capacity.

Study-note section NET24

> A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

> Address Space: An address space is the total
> number ofaddresses used by the protocol.
> • If a protocol uses N bits to define an address,
> the address space is 2N
> because each bit can
> have twodifferent values (0 or 1) and N bits
> can have 2N values.
> • IPv4 uses 32-bit addresses, which means that
> the address space is 232 or 4,294,967,296
> (more than 4billion).
> Notations: There are two notations to show an IPv4 address:
> • Binary notation: The IPv4 address is displayed
> as 32bits. ex. 11000001 10000011 00011011
> 11111111
> • Dotted decimal notation: To make the IPv4
> address easier to read, Internet addresses are
> usually written in decimal form with a decimal

point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Mock exam 2 · Question 21

Question 151 | SM0021

Which one of the following is NOT correct about POP3 and SMTP?

A. A sender can send email using SMTP

B. Both are application layer protocols

C. A sender can receive emails using POP3

D. Both are network layer protocols

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: Both are network layer protocols

The application-layer list includes SMTP and POP3; POP3 is explicitly for receiving mail. They are not network-layer protocols. Evidence: PDF 28, 18.

Why each choice is right or wrong

A - Incorrect: SMTP mail transfer supports sending (PDF 28, 18.)

B - Incorrect: Both appear in the application list (PDF 28, 18.)

C - Incorrect: Receiving via POP3 matches the explicit receive-mail statement (PDF 28, 18.)

D - Correct: Calling both network-layer protocols contradicts the layer list (PDF 28, 18.)

Primary note proof

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the
combined session, presentation, and application layers in
the OSI model and many protocols are defined at this
layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Question sources

MOE Computer Science Exit Exam · Question 6-1

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 143

Question 152 | SM0022

A university’s private LANs and WANs provide services only to its staff, students and administrators. Which term best describes this system by its access scope?

A. Local Area Network

B. Intranet

C. Wide Area Network

D. Metropolitan Area Network

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B: Intranet

The relevant distinction is who can access organizational services, rather than how far the links extend. An intranet serves an organization internally, so it can include both local and wide-area components. LAN, WAN and MAN describe geographic reach and cannot replace that access-scope classification. The option identifies an intranet without assuming a particular geographic footprint.

Why each choice is right or wrong

A - Incorrect: LAN describes geographic reach of local components rather than the organizational access scope requested.

B - Correct: An intranet serves the university’s internal members across its constituent networks.

C - Incorrect: WAN describes geographic reach of some components rather than the internal-access scope requested.

D - Incorrect: MAN describes metropolitan geographic reach rather than internal organizational access.

Primary note proof

Study-note section NET02

> PAN connects personal devices, such as Bluetooth accessories. LAN covers a local building or campus; MAN spans a metropolitan region; WAN connects distant regions. Device count alone does not determine the category. The Internet is a global network of networks, commonly the largest example of a WAN.

Study-note section NET02

> An intranet serves an organization internally; an extranet exposes selected services to approved outsiders. Neither term alone specifies geographic size. A client requests a service; a server

supplies it. Peer systems may perform both roles. Dedicated servers concentrate on service workloads but can still be clients of DNS, backup or update services. A ten-user cutoff is a teaching guideline, not a technical law. Centralization simplifies administration but creates dependencies; peer networks still need security.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 26-44

Personal Area Network (PAN): is a network may be spanned across a given table with distances between the devices not more than few meters. The technology used to interconnect the devices could be a Bluetooth. These networks are called Personal Area Networks, since the devices interconnected in these networks are belongs to a single person.
Local Area Network (LAN): is a network that may span across a building, or across several buildings within a single organization, by using intermediate devices, like switches and/or hubs, to interconnect devices in all floors.
Metropolitan Area Network (MAN): is a network that may span across a whole city interconnecting several buildings and organizations.
Wide Area Network (WAN): is a network that may span across multiple cities, or an entire country, or an entire continent or it may even cover the whole world. For example, an Internet is one example of WAN.

Data Communication and Computer Networking module | PDF page 24 | lines 23-35

Network Architecture: based on the architecture (where do the clients get the shared resources?), networks can be categorized into three:
Client-Server Architecture: There can be one or more systems acting as Server. Other being Client, request the Server to serve requests. Servers take and process request on clients‘ behalf.
Peer-to-Peer (Point-to-point): Two systems can be connected Point-to-Point, or in other words back-to-back fashion. They both reside on same level and called peers.
There can be hybrid network which involves network architecture of both the above types.

Question sources

MOE Computer Science Exit Exam · Question 6-2

National Computer Science Exit Exam · Question 78

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 144

Question 153 | SM0025

At which layer of the OSI model are devices such as bridges, switches and Network Interface Cards used?

A. Physical layer

B. Application layer

C. Network layer

D. Data link layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Data-link forwarding functions, with NICs also implementing PHY

Bridges and ordinary switches inspect MAC frames at layer 2. A NIC commonly implements both physical and data-link functions, as NET09 explains; listing NICs here does not force all NIC work into one layer. D is the common frame/addressing layer for the listed devices' relevant network-interface forwarding functions.

Why each choice is right or wrong

A - Incorrect: A NIC has physical functions, but bridges/switches make layer-2 forwarding decisions beyond signal repetition.

B - Incorrect: These devices' MAC/frame operations are below application-service semantics.

C - Incorrect: Ordinary bridges and Ethernet switches do not route IP prefixes as their defining function.

D - Correct: Bridges/switches operate at layer 2, and NICs also implement link-layer MAC/frame functions.

Primary note proof

Study-note section NET09

> A NIC supplies a computer's network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Study-note section NET10

A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets
across anetwork.
Unlike a hub which simply replicates what it receives on
oneport onto all the other ports, a switch keeps a record
of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the
packetdirectly to the recipient device by looking up the
MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 12 | lines 60-61

BRIDGE: Connects two LANS having the same protocol -
(e.g. Ethernet or Token ring).

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to
resolve thephysical address of a device on a network,
where its logical address is known.
Physical address is the 48 bit address that is imprinted
on theNIC or LAN card, Logical address is the Internet
Address orcommonly known as IP address that is used
to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is
used bya device on the network to find its Internet
address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical
layer by providing error detection and correction
mechanisms.
• Framing: the Data Link layer receives the data
from Network Layer and divides the stream of bits
into fixed size manageable units called as Frames
and sends it to the physical layer. On the receiver

side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Question sources

MOE Computer Science Exit Exam · Question 7-2

National Computer Science Exit Exam · Question 39

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 145

Question 154 | SM0035

Which listed device is an end resource used by clients, rather than an intermediary that forwards, bridges or repeats network traffic?

A. Routers

B. Network printer

C. Wireless access point

D. Hubs

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B: Network printer

A network printer is an endpoint resource that receives client jobs. A router forwards packets between networks, a wireless access point normally bridges wireless and wired link traffic, and a hub repeats physical signals among ports. These are intermediary roles in the comparison. The question classifies their normal function rather than claiming that an intermediary cannot also expose a management endpoint.

Why each choice is right or wrong

A - Incorrect: Routers forward traffic between networks and are intermediaries.

B - Correct: A network printer is the client-facing end resource.

C - Incorrect: A wireless access point normally bridges wireless and wired link traffic.

D - Incorrect: A hub repeats signals between attached devices and is an intermediary.

Primary note proof

Study-note section NET09

> A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET-S04

> A computer network connects computers or other devices so they can exchange information and share resources such as files, storage and printers. The interconnection uses both hardware and software; neither hardware alone nor software alone is the definition of the whole network.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

> ROUTER: Router is a specialized network device
> used tointerconnect different types of network that
> uses different protocol.
> Router allows the users to connect several LAN and WAN
> • A LAN to another LAN
> • A WAN to another WAN
> • A LAN to the Internet

Routers use the routing table to determine how to forwardpackets.
• The list of routing tables provides directions to transfer the data to a particular network destination -various paths to forward the data.
• A router analyzes a destination IP address of a givenpacket and compares it with the routing table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is normallyused to create connections between stations in a physical startopology.
HUB: Hubs are commonly used to connect segments (devices)of a LAN. Hubs don‘t filter information; instead broadcast incoming packets to all computers connected to it. A hub contains multiple ports which support 8, 12 or 24 RJ-45 ports. Two types: Passive hub and Active hub. It is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 22 | lines 26-44

Personal Area Network (PAN): is a network may be spanned across a given table with distances between the devices not more than few meters. The technology used to interconnect the devices could be a Bluetooth. These networks are called Personal Area Networks, since the devices interconnected in these networks are belongs to a single person.
Local Area Network (LAN): is a network that may span across a building, or across several buildings within a single organization, by using intermediate devices, like switches and/or hubs, to interconnect devices in all floors.
Metropolitan Area Network (MAN): is a network that may span across a whole city interconnecting several buildings and organizations.
Wide Area Network (WAN): is a network that may span across multiple cities, or an entire country, or an entire continent or it may even cover the whole world. For example, an Internet is one example of WAN.

Data Communication and Computer Networking module | PDF page 22 | lines 2-19

A network can be defined as two or more computers connected together in such a way that they can share resources. The primary purpose of a network is to share

resources, and a resource could be:
• a file,
• a folder,
• a printer,
• a disk drive, or just about anything else that exists on a computer.
Therefore, a computer network is simply a collection of computers or other hardware devices that are connected together, either physically or logically, using special hardware and software, to allow them to exchange information and cooperate. Networking is the term that describes the processes involved in designing, implementing, upgrading, managing and otherwise working with networks and network technologies.

Question sources

MOE Computer Science Exit Exam · Question 11-1

National Computer Science Exit Exam · Question 15

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 146

Question 155 | SM0036

_____ is a network architecture characteristic that states a network can expand quickly to address needs of new users and applications without having impact on the services delivered to existing users.

A. Security

B. Quality of service

C. Fault tolerance

D. Scalability

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer D: Scalability

Scalability is explicitly continued predictable function as system size grows, matching expansion without undermining existing service. Evidence: PDF 32.

Why each choice is right or wrong

A - Incorrect: Security concerns authorized access, not size growth (PDF 32.)

B - Incorrect: Quality of service is not the named growth principle (PDF 32.)

C - Incorrect: Fault tolerance is continued operation through failures rather than growth (PDF 32.)

D - Correct: Scalability is the explicit size-growth principle (PDF 32.)

Primary note proof

Study-note section NET01

> Delivery means reaching the intended recipient; accuracy means preserving content; timeliness means meeting a deadline; jitter means variation in arrival delay. A call can have acceptable average delay yet sound broken because of jitter. Throughput measures achieved transfer rate, reliability measures dependable operation, and security concerns unauthorized actions. Scalability means accommodating growth without unacceptable service degradation.

Study-note section NET-S02

> Quality of service (QoS) controls traffic treatment to meet objectives such as throughput, delay, jitter or loss. Classification, queue scheduling and allocation of buffer or bandwidth resources implement such treatment. QoS concerns delivered service behavior; scalability concerns accommodating growth, and security concerns unauthorized actions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 32 | lines 2-5

> Scalable systems are those that grow in
> accordance with policy; i.e. they continue to
> function predictably, even as they increase in
> size.

Question sources

MOE Computer Science Exit Exam · Question 11-2

National Computer Science Exit Exam · Question 17

National Computer Science Exit Exam · Question 19

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 147

Question 156 | SM0038

Which network type is the largest as compared to the rest?

A. Wide Area Network

B. Metropolitan Area Network

C. The Internet

D. Local Area Network

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: The Internet

The Internet names the particular global network of networks, while WAN, MAN and LAN name scope categories. NET02 explicitly gives the Internet as the largest common WAN example. Thus C is the intended largest actual network in the comparison, even though WAN includes Internet-scale networks rather than being a disjoint smaller category.

Why each choice is right or wrong

A - Incorrect: WAN is a category that includes the Internet, not a separate specific network shown to be larger.

B - Incorrect: MAN spans a metropolitan area rather than the global network of networks.

C - Correct: NET02 identifies the global Internet as the largest common WAN example.

D - Incorrect: LAN covers a local building/campus and is much narrower in geographic scope.

Primary note proof

Study-note section NET02

> PAN connects personal devices, such as Bluetooth accessories. LAN covers a local building or campus; MAN spans a metropolitan region; WAN connects distant regions. Device count alone does not determine the category. The Internet is a global network of networks, commonly the largest example of a WAN.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 26-44

> Personal Area Network (PAN): is a network may be spanned across a given table with distances between the devices not more than few meters. The technology used to interconnect the devices could be a Bluetooth. These networks are called Personal Area Networks, since the devices interconnected in these networks are belongs to a single person.

Local Area Network (LAN): is a network that may span across a building, or across several buildings within a single organization, by using intermediate devices, like switches and/or hubs, to interconnect devices in all floors.
Metropolitan Area Network (MAN): is a network that may span across a whole city interconnecting several buildings and organizations.
Wide Area Network (WAN): is a network that may span across multiple cities, or an entire country, or an entire continent or it may even cover the whole world. For example, an Internet is one example of WAN.

Question sources

MOE Computer Science Exit Exam · Question 11-4

National Computer Science Exit Exam · Question 88

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 148

Question 157 | SM0057

Which statement is incorrect when packet switching means connectionless datagram switching?

A. In packet switching, messages are sent in small blocks

B. In packet switching, an end-to-end connection has to be established

C. In circuit switching, a channel is dedicatedly used

D. For bursty traffic, statistical packet sharing can use capacity that dedicated circuits would reserve while idle.

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer B: In packet switching, an end-to-end connection has to be established

Datagram switching divides messages into packets and forwards those packets independently without requiring end-to-end connection establishment. Circuit switching reserves resources along a path. Statistical sharing lets bursty packet traffic use capacity that would otherwise remain reserved but idle; this is a conditional utilization advantage, not a universal claim that packets outperform circuits for every workload or metric.

Why each choice is right or wrong

A - Incorrect: Packet switching divides messages into smaller packets.

B - Correct: Connectionless datagrams do not require end-to-end connection setup.

C - Incorrect: Circuit switching reserves channel resources for the connection.

D - Incorrect: For bursty traffic, shared packet links can avoid idle dedicated reservations; the qualified statement is true.

Primary note proof

Study-note section NET31

Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Study-note section NET31

Datagram packets are forwarded independently. Virtual circuits establish forwarding state, but resource reservation is optional, not a necessary property of every virtual circuit. Message switching stores a complete message before forwarding, without a dedicated end-to-end path. Packet store-and-forward waits for the full packet/frame; cut-through starts after enough header arrives; fragment-free historically waits through the initial collision-sensitive portion. No routing algorithm guarantees that every packet always takes the geometrically shortest route.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 20 | lines 12-26

Circuit Switching: In circuit switching, two communicating stations are connected by a dedicated communication path which consists of intermediate nodes in the network and the links that connect these nodes.
It involved the following three distinct steps
• Circuit Establishment: To establish an end-to-end connection before any transfer of data.
• Data transfer: Transfer data is from the source to the destination. The data may be analog or digital, depending on the nature of the network. The connection is generally full-duplex.
• Circuit disconnect: Terminate connection at the end of data transfer. Signals must be propagated to deallocate the dedicated resources.

Data Communication and Computer Networking module | PDF page 20 | lines 27-36

Message Switching:
Packet Switching: Packet switching was designed to

address the shortcomings of circuit switching in dealing with data communication. Unlike circuit switching where communication is continuous along a dedicated circuit, in packet switching, communication is discrete in form of packets.
The three switching modes to choose from are:
• Cut-through
• Store-and-forward
• Fragment-free

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Question sources

MOE Computer Science Exit Exam · Question 16-4

National Computer Science Exit Exam · Question 97

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 149

Question 158 | SM0062

Which layer uses port number to identify applications?

A. Transport layer

B. Physical layer

C. Network layer

D. Application layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: Transport layer

Ports explicitly identify source and destination processes for transport delivery. Evidence: PDF 16, 28, 14, 15, 16.

Why each choice is right or wrong

A - Correct: Transport uses port addresses for processes (PDF 16, 28, 14, 15, 16.)

B - Incorrect: Physical represents bits and signals (PDF 16, 28, 14, 15, 16.)

C - Incorrect: Network uses logical IP addresses (PDF 16, 28, 14, 15, 16.)

D - Incorrect: Application uses transport services but port addressing is assigned to transport (PDF 16, 28, 14, 15, 16.)

Primary note proof

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 28 | lines 2-14

TCP and UDP must use port numbers to communicate with the upper layers, because they‘re what keep track of different conversations crossing the network simultaneously. These port numbers identify the source and destination application or process in the TCP segment. There are 216 = 65,536 ports available.
Well-known ports: The port numbers range from 0 to 1,023.
Registered ports: The port numbers range from 1,024 to 49,151. Registered ports are used by applications or services that need to have consistent port assignments.
Dynamic or private ports: The port numbers range from 49,152 to 65,535.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.

• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Question sources

MOE Computer Science Exit Exam · Question 18-1

Computer Science Exit Exam practice collection · Question 150

Question 159 | SM0225

In which network topology does a single line cable called a backbone connect all nodes on a network without intervening connected devices?

A. Bus

B. Ring

C. Star

D. Mesh

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer A: Bus

Bus explicitly connects all devices to a central backbone cable terminated at both ends. Evidence: PDF 23, 23, 23, 23.

Why each choice is right or wrong

A - Correct: Bus has the stated shared backbone (PDF 23, 23, 23, 23.)

B - Incorrect: Ring is a closed neighbor loop (PDF 23, 23, 23, 23.)

C - Incorrect: Star uses a central hub/switch and separate links (PDF 23, 23, 23, 23.)

D - Incorrect: Mesh has many direct node-pair links (PDF 23, 23, 23, 23.)

Primary note proof

Study-note section NET08

> Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Study-note section NET08

> Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 4-18

> Bus Topology: in this topology all devices are
> connected to a central cable, called the bus or
> backbone, which is terminated at its ends. The

purpose of the terminators is to stop the signal from bouncing, thereby clearing the cable so that other computers can send data. Message transmitted along the Bus is visible to all computers connected to the backbone cable. As the message arrives at each workstation, the workstation checks the destination address contained in the message either to process or drop the packet if it matches or not respectively. Its advantages are, ease of installation and less amount of cable requirement. Its main drawback is, the entire network will be shut down if there is a break in the main cable.

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Data Communication and Computer Networking module | PDF page 23 | lines 35-44

Ring Topology: in this topology, all devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it. Some of its advantages include, easy to install and reconfigure, less expensive (than mesh), and performance is even despite the number of users. Its cons include, break in the ring (such as a disabled station) can disable the entire network, and limitations on media and traffic (limitation on ring length and number of devices).

Data Communication and Computer Networking module | PDF page 23 | lines 45-53

Mesh Topology: in this topology devices are connected with many redundant interconnections

between network nodes. In a full mesh topology, every node has a connection to every other node in the network, which makes it the most expensive topology over all the other topologies. The number of cables grows fast as the number of nodes increases, and it can be calculated by using the general formula ((n (n - 1)) /2), where n is the number of nodes in the

Question sources

Computer Science Exit Exam · Sene 2016 · Question 8-3

Computer Science Exit Exam practice collection · Question 158

Question 160 | SM0233

In computer networking, _____ refers to how various nodes, devices, and connections on your network are physically or logically arranged in relation to each other.

A. Network layer

B. Network installation

C. Network protocol

D. Network topology

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer D: Network topology

Network topology is the shape/layout and geometrical relation of links and devices, including physical and logical connectedness. Evidence: PDF 22, 5.

Why each choice is right or wrong

A - Incorrect: Network layer is a protocol responsibility layer (PDF 22, 5.)

B - Incorrect: Installation is the activity of building, not the layout name (PDF 22, 5.)

C - Incorrect: Protocol is a communication rule set (PDF 22, 5.)

D - Correct: Topology is exactly the arrangement concept (PDF 22, 5.)

Primary note proof

Study-note section NET08

Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Study-note section NET03

Protocol syntax defines message structure, semantics defines field meanings, and timing defines when/how fast exchanges occur. A standard supports interoperability. De facto means established through use; de jure means formally approved. ISO develops international standards; IEEE develops LAN standards such as 802.3 and 802.11; ITU-T develops telecommunications standards. ANSI coordinates US standards, EIA is associated with interface standards, and FCC is a regulator.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 22 | lines 45-52

Inter-connectivity: components of a network,
including end devices and interconnecting devices,
can be connected to each other differently in some
fashion. By connectedness we mean either logically,
physically or both ways. Network topology refers to
the shape of a network, or the network‘s layout. It is
the geometric representation of the relationship of all
the links and linking devices to one another. There are

Data Communication and Computer Networking module | PDF page 5 | lines 70-81

PROTOCOL
Protocol can be defined as set of rules that governs data
communications.
It defines what is to be communicated, how it is to be
communicated and when it is to be communicated.
There are three key elements of a protocol:
Syntax: It means the structure or format of the data. It is the
arrangement of data in a particular order.
Semantics: It tells the meaning of each section of bits and
indicates the interpretation of each section.
Timing: It tells the sender about the readiness of the receiver to
receive the data

Question sources

Computer Science Exit Exam · Sene 2016 · Question 11-1

Computer Science Exit Exam practice collection · Question 163

Question 161 | SM0264

Which one of the following IS NOT a component of communication?

A. Distance

B. Source

C. Destination

D. Medium

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: Distance

The five components are message, sender, receiver, medium and protocol. Distance is a property of a path, not a component in this definition. Evidence: PDF 5.

Why each choice is right or wrong

A - Correct: Distance is not one of the five defined components (PDF 5.)

B - Incorrect: Source corresponds to sender (PDF 5.)

C - Incorrect: Destination corresponds to receiver (PDF 5.)

D - Incorrect: Medium is explicitly listed (PDF 5.)

Primary note proof

Study-note section NET01

> Data are raw representations; information is data interpreted for a purpose. Communication requires a message, sender, receiver, medium and agreed protocol. A source creates data; a transmitter converts it to a signal; a receiver reconstructs data. Distance affects engineering but is not a sixth component.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 5 | lines 23-35

> A Data Communication system has five components:-
> • Message:- is the information to be communicated by the sender to the receiver.
> • Sender The sender is any device that is capable of sending the data (message).
> • Receiver: - is a device that the sender wants to communicate the data (message).
> • Transmission Medium: - It is the path by which the message travels from sender to receiver. It can be wired or wireless and many subtypes in both.

• Protocol:- It is an agreed upon set or rules used by the sender and receiver to communicate data. Rules that govern data communication.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 21-1

Computer Science Exit Exam practice collection · Question 162

Question 162 | SM0274

In which network topology should every computer have N-1 I/O ports when there are N computers in the network?

A. Star

B. Mesh

C. Bus

D. Ring

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer B: Mesh

In full mesh every node directly connects to each of the other N-1 nodes. With a separate port for each dedicated link, each computer needs N-1 ports. Evidence: PDF 23, 23, 23, 23.

Why each choice is right or wrong

A - Incorrect: Star needs one link per end device to the center (PDF 23, 23, 23, 23.)

B - Correct: Full mesh has one connection to each other node (PDF 23, 23, 23, 23.)

C - Incorrect: Bus shares a backbone rather than N-1 separate links (PDF 23, 23, 23, 23.)

D - Incorrect: Ring connects each node to two neighbors (PDF 23, 23, 23, 23.)

Primary note proof

Study-note section NET08

A full undirected mesh of n nodes needs n(n-1)/2 links and n-1 interfaces per node. For 50 nodes: 50×49/2=1225 links. Redundant paths improve fault tolerance at greater cable and interface cost.

Study-note section NET08

Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while

protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 45-53

Mesh Topology: in this topology devices are connected with many redundant interconnections between network nodes. In a full mesh topology, every node has a connection to every other node in the network, which makes it the most expensive topology over all the other topologies. The number of cables grows fast as the number of nodes increases, and it can be calculated by using the general formula ((n (n - 1)) /2), where n is the number of nodes in the

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Data Communication and Computer Networking module | PDF page 23 | lines 35-44

Ring Topology: in this topology, all devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it. Some of its advantages include, easy to install and reconfigure, less expensive (than mesh), and performance is even despite the number of users. Its cons include, break in the ring (such as a disabled station) can disable the entire network, and limitations on media and traffic (limitation on ring length and number of devices).

Data Communication and Computer Networking module | PDF page 23 | lines 4-18

Bus Topology: in this topology all devices are connected to a central cable, called the bus or backbone, which is terminated at its ends. The purpose of the terminators is to stop the signal from bouncing, thereby clearing the cable so that other computers can send data. Message transmitted along the Bus is visible to all computers connected to the backbone cable. As the message arrives at each workstation, the workstation checks the destination address contained in the message either to process or drop the packet if it matches or not respectively. Its advantages are, ease of installation and less amount of cable requirement. Its main drawback is, the entire network will be shut down if there is a break in the main cable.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 24-2

Computer Science Exit Exam practice collection · Question 159

Question 163 | SM0291

In data communication, data rate refers how fast we can send data and is measured in bits per second. Which one of
the following is not a factor for data rate?

A. Travel distance

B. Levels of the signals

[SS

C. Bandwidth available

D. Level of noise

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: Travel distance is not an explicit channel-capacity formula variable

NET05 gives Nyquist in terms of bandwidth and signal states, and Shannon in terms of bandwidth and signal-to-noise ratio. Distance is not an independent variable in those simplified capacity formulas, so A is the intended comparison. It can still affect practical achievable rate through attenuation and delay. The image confirms travel distance despite noise in another extracted option.

Why each choice is right or wrong

A - Correct: Distance is absent as an independent variable in the simplified formulas, although it affects real links indirectly.

B - Incorrect: The number of signal levels appears explicitly in Nyquist through log2(M).

C - Incorrect: Available bandwidth directly appears as B in both Nyquist and Shannon capacity expressions.

D - Incorrect: Noise affects signal-to-noise ratio and therefore the Shannon capacity limit.

Primary note proof

Study-note section NET05

For an ideal noiseless channel, Nyquist gives R≤2B log2(M). For Gaussian noise, Shannon gives C=B log2(1+S/N), with a linear power ratio; SNR(dB)=10 log10(S/N). Bandwidth, signal levels and noise appear explicitly. Distance affects attenuation, delay and often achievable rate, but is not a separate variable in these simplified capacity formulas.

Study-note section NET06

Transmission delay is packet bits divided by link bit rate; propagation delay is distance divided by signal speed. Processing and queueing add delay. A 1000-byte packet takes 0.8 ms to serialize at 10 Mbit/s; increasing line rate does not reduce propagation time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 9 | lines 2-14

BIT LENGTH or Bit Interval (Tb)
It is the time required to send one bit.
It is measured in seconds.
BIT RATE
It is the number of bits transmitted in one second.
It is expressed as bits per second (bps).
Relation between bit rate and bit interval can be as follows
Bit rate = 1 / Bit interval
Baud Rate
It is the rate of Signal Speed, i.e the rate at which the signal
changes.
A digital signal with two levels 0_& 1_will have the same
baud rate and bit rate & bit rate.

Data Communication and Computer Networking module | PDF page 10 | lines 18-35

receiver.There are three causes of impairment:
• Attenuation
• Distortion
• Noise
Attenuation
Attenuation results in loss of energy due to distance.

Whena signal travels through a medium, it loses some of its energy in overcoming the resistance of the medium. The electrical energy in the signal may convert to heat. To compensate for this loss, amplifiers are used to amplify thesignal.
Distortion
Distortion changes the shape of the signal.
Noise
Noise is any unwanted signal that is mixed or combined with the original signal during transmission. Due to noise the original signal is altered and signal received is not sameas the one sent.

Data Communication and Computer Networking module | PDF page 9 | lines 29-49

Transmission of Digital signal: Digital signal can be transmitted in the following two ways:
• Baseband Transmission: The signal is transmitted without making any change to it (i.e. Without modulation)
o In baseband transmission, the bandwidth of the signal to be transmitted has to be less than the bandwidth of the channel.
o Ex. Consider a Baseband channel with lower frequency 0Hz and higher frequency 100Hz, hence its bandwidth is 100 (Bandwidth is calculated by getting the difference between the highest and lowest frequency)
o A channel whose bandwidth is more than the bandwidth of the signal is called Wideband channel
o A channel whose bandwidth is less than the bandwidth of the signal is called Narrowband channel
• Broadband Transmission: Given a bandpass channel, a digital signal cannot be transmitted directly through it
o In broadband transmission we use modulation, i.e. we change the signal to analog signal before transmitting it.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 30-1

Question 164 | SM0292

Which layer of the OSI model ensures that the whole message (group of packets) arrives at the destination intact and in order?

A. Session layer

B. Transport layer

C. Data link layer

D. Network layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Transport layer

Transport explicitly makes process-delivered data intact and ordered, using sequence-numbered segments and end-to-end error control. Evidence: PDF 16, 15, 15, 16.

Why each choice is right or wrong

A - Incorrect: Session controls dialogs/checkpoints (PDF 16, 15, 15, 16.)

B - Correct: Transport explicitly ensures intact and ordered whole-message delivery (PDF 16, 15, 15, 16.)

C - Incorrect: Data link controls hop-to-hop frames rather than the entire end-to-end message (PDF 16, 15, 15, 16.)

D - Incorrect: Network packets can arrive out of order (PDF 16, 15, 15, 16.)

Primary note proof

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Study-note section NET19

OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.

• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 16 | lines 18-32

Session Layer
• The session layer establishes a session between the communicating devices called dialog and synchronizes their interaction. It is the responsibility of the session layer to establish and synchronize the dialogs. It is also called the network dialog controller.
• The session layer at the sending side accepts data from the presentation layer adds checkpoints to it called sync bits and passes the data to the transport layer. At the receiving end the session layer receives

data from the transport layer removes the checkpoints inserted previously and passes the data to the presentation layer. So, the main responsibility of session layer is dialog control and synchronization.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 30-2

Computer Science Exit Exam practice collection · Question 165

Question 165 | SS098

Which of the following is not a valid host address within the network 172.17.128.0/21 ?

A. 172.17.128.1/21

B. 172.17.135.255 /21

C. 172.17.135.0 /21

D. 172.17.128.255/21

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: 172.17.135.255/21 is broadcast

A /21 groups eight third-octet values, here 128-135. The network is 172.17.128.0 and its broadcast is 172.17.135.255. B sets every host bit to one and cannot be assigned to an ordinary host. The other choices are inside the usable range; ending in .0 or .255 alone does not make an address reserved.

Why each choice is right or wrong

A - Incorrect: 172.17.128.1 is one above the network boundary and is a valid usable host.

B - Correct: 172.17.135.255 sets all eleven host bits to one and is the subnet broadcast address.

C - Incorrect: 172.17.135.0 is inside the /21 block; its host bits are not all zero or all one.

D - Incorrect: 172.17.128.255 is an interior host; the .255 ending alone does not make it /21 broadcast.

Primary note proof

Study-note section NET25

For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Study-note section NET25

192.168.2.64/26 spans .64-.127; usable .65-.126 gives 62 hosts. 192.168.10.25/28 spans .16-.31, so its network is .16. 172.17.128.0/21 spans 172.17.128.0-172.17.135.255; .128.255 and .135.0 are valid interior hosts, but .135.255 is broadcast. /27 gives 32 total and 30 usable, not 32 usable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP

addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Question sources

Computer Science model exam collection · Question 5

Question 166 | SS111

Which of the following is not the function of Cross-over UTP cable?

A. To connect router to pc

B. To connect switch to switch

C. To connect switch to pc

D. To connect pc to pc

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer C: Switch-to-PC normally uses straight-through cable in legacy Ethernet

Under the legacy MDI/MDI-X convention, a switch port and PC have unlike roles and use straight-through cable. PCs and router Ethernet endpoints conventionally share the endpoint role, while two switches share the switch role, so those like-role links use crossover. Auto-MDI-X can remove the practical requirement; the question is testing the conventional cable-role distinction.

Why each choice is right or wrong

A - Incorrect: A legacy router Ethernet endpoint and PC have like interface roles, so crossover is appropriate.

B - Incorrect: Two legacy switch ports have like roles and conventionally require crossover.

C - Correct: Switch and PC have unlike legacy interface roles and normally use straight-through cable.

D - Incorrect: Two PCs have like legacy interface roles; NET07 explicitly names PC-PC as a crossover example.

Primary note proof

Study-note section NET07

Radio, microwave and infrared are unguided media. Ground waves follow Earth, sky waves can reflect from ionized layers, and line-of-sight paths require suitable visibility. Microwaves are radio waves in the broader physical classification. Infrared generally does not penetrate walls. Directionality depends on antennas: omnidirectional means broad azimuth coverage, not equal radiation in every 3-D direction. Legacy Ethernet straight-through cables join unlike MDI/MDI-X roles; crossover joins like roles, such as PC-PC or switch-switch. Auto-MDI-X can remove that requirement.

Study-note section NET-S07

For legacy fixed Ethernet port roles, PCs and router Ethernet endpoints use the same endpoint role, while ordinary switch ports use the opposite role. Thus a direct router-to-PC link uses crossover wiring; PC-to-switch or router-to-switch uses straight-through wiring. Automatic MDI-X can remove the need to choose the crossover manually.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 11 | lines 23-33

Twisted-pair cable: Consists of two conductors (normally copper), each with its own plastic insulation, twisted together.
Two types
• Unshielded Twisted-Pair cable (UTP)
• Shielded Twisted-Pair cable (STP)
Coaxial cable:- uses copper wire to conduct the signals electronically
Fiber-optic cable:- uses glass or plastic conductor and transmits the signals as light
Wireless/Unguided media the signal propagates in the atmosphere and in outer space

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more

cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Question sources

Computer Science model exam collection · Question 18

Question 167 | SS126

Given a situation two hosts attempt on a half-duplex Ethernet LAN to send data concurrently, resulting in a collision and subsequent to this collision,
what will the hosts do?

A. An electrical pulse indicates that the collision has cleared

B. The router on the segment will signal that the collision has cleared

C. The jam signal indicates that the collision has been cleared

D. The hosts will attempt to resume transmission after a time delay has expired

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer D: Stop and retry after random backoff

NET14 gives the collision sequence: detect, jam, stop and retry after random backoff. The jam indicates the collision itself, not that it has cleared. There is no router-coordinated clearance procedure or electrical clearance pulse in the stated rule. Therefore waiting for a delay and attempting transmission again is the supported host behavior.

Why each choice is right or wrong

A - Incorrect: The defined procedure uses collision indication and backoff; it does not include a collision-cleared electrical pulse.

B - Incorrect: Ethernet collision recovery is performed by the hosts' MAC procedure, not a router clearance signal.

C - Incorrect: A jam announces the collision; it does not certify that the shared medium is clear.

D - Correct: After stopping, hosts retry following random backoff, matching a delay before another transmission attempt.

Primary note proof

Study-note section NET14

Random access allows contention: ALOHA sends opportunistically; CSMA senses before sending. In legacy half-duplex Ethernet, CSMA/CD detects collision, sends a jam indication, stops, and retries after random backoff. The jam signals a collision, not its clearance. Wi-Fi 802.11 uses

CSMA/CA with waiting/backoff and acknowledgments; sensing does not eliminate hidden-terminal collisions.

Study-note section NET14

Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 21 | lines 6-26

MEDIUM ACCESS CONTROL
The problem of controlling the access to the medium is similarto the rules of speaking in a meeting (the right to speak isupheld; two people do not speak at the same time; do notinterrupt each other; do not monopolize the discussion; ...) Three categories of multiple access protocols
Random Access Protocols - try your best like taxis do
• MA - Multiple Access
• CSMA - Carrier Sense MA
• CSMA/CD - CSMA with Collision Detection
• CSMA/CA - CSMA with Collision AvoidanceControlled-Access Protocols - get permission
• Reservation
• Polling
• Token Passing
Channelization Protocols - simultaneous use
• FDMA - Frequency-Division MA
• TDMA - Time-Division MA
• CDMA - Code-Division MA

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is normallyused to create connections between stations in a physical startopology.
HUB: Hubs are commonly used to connect segments (devices)of a LAN. Hubs don‘t filter information; instead broadcast incoming packets to all computers connected to it. A hub contains multiple ports which support 8, 12 or

24 RJ-45 ports. Two types: Passive hub and Active hub. It is used in star or ring topology.

Question sources

Computer Science model exam collection · Question 33

Question 168 | SS134

Which of the following is true regarding the use of switches and hubs for network connectivity in the network?

A. Switches increase the number of collision domains in the network

B. Switches take less time to process frames than hubs take

C. Switches do not forward broadcasts

D. Hubs can filter frames

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer A: Switch ports separate collision domains

NET10 says each switch port separates collision domains, whereas a half-duplex hub repeats traffic into one shared collision domain. Switches still flood broadcasts within their VLAN, and hubs do not inspect and filter destination frames. The note gives no universal switch-versus-hub processing-time advantage, so A is the supported claim.

Why each choice is right or wrong

A - Correct: Replacing a shared hub segment with separate switch ports divides the network into additional collision domains.

B - Incorrect: The note establishes no universal lower switch processing time; a hub simply repeats signals.

C - Incorrect: Switches flood broadcasts within the VLAN unless another control filters them.

D - Incorrect: Hubs repeat physical signals and neither learn MAC destinations nor filter frames.

Primary note proof

Study-note section NET10

Each switch port separates collision domains; full-duplex point-to-point Ethernet has no collisions and does not use CSMA/CD. Switching permits independent conversations simultaneously. Broadcasts still reach the VLAN unless another control filters them. A layer-3 switch also routes, so device marketing names do not override the function being examined.

Study-note section NET09

A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is normallyused to create connections between stations in a physical startopology.
HUB: Hubs are commonly used to connect segments (devices)of a LAN. Hubs don‘t filter information; instead broadcast incoming packets to all computers connected to it. A hub contains multiple ports which support 8, 12 or 24 RJ-45 ports. Two types: Passive hub and Active hub. It is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets across anetwork.
Unlike a hub which simply replicates what it receives on oneport onto all the other ports, a switch keeps a record of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the packetdirectly to the recipient device by looking up the MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 13 | lines 2-8

Filters content by reading the MAC addresses of source and destination.
• MAC to Segment number table
• Do not filter broadcast
• Data not destined for other network is prevented from passing over the bridge
• Slower than repeater due to filtering

Question sources

Computer Science model exam collection · Question 41

Question 169 | SS145

A TCP/IP layer which is responsible for addressing, path selection and routing refers:

A. Transport Layer

B. Network Access Layer

C. Application Layer

D. Internet Layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Internet Layer

The TCP/IP Internet layer corresponds to OSI network and provides IP addressing/source-to-destination transmission with routing. Evidence: PDF 17, 15, 17.

Why each choice is right or wrong

A - Incorrect: Transport is process delivery (PDF 17, 15, 17.)

B - Incorrect: Network access combines physical/data link (PDF 17, 15, 17.)

C - Incorrect: Application combines OSI upper layers (PDF 17, 15, 17.)

D - Correct: Internet matches network-layer addressing and routing (PDF 17, 15, 17.)

Primary note proof

Study-note section NET19

> The four-layer TCP/IP model folds OSI layers 5-7 into application, retains transport, calls layer 3 Internet, and combines link/physical as network access. A model is a guide to functions, not a guarantee that every application gets a reliable network. Distinguish end-to-end application service from its underlying process-to-process transport.

Study-note section NET30

> Routing computes paths; forwarding chooses a next hop for an individual packet. Select the matching route with the longest prefix, then apply the routing system's tie-breaking criteria. A default /0 matches when nothing more specific does. Static routes are configured; dynamic protocols exchange reachability. Distance vector uses neighbor estimates, while link state distributes link information and computes shortest paths. Interior protocols operate within an autonomous system; BGP exchanges routes between systems.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 22-38

> Network Layer or IP
> Also called as the Internetwork Layer (IP). It holds the

IP protocol which is a network layer protocol and is responsiblefor source to destination transmission of data.
The Internetworking Protocol (IP) is a connection-less & unreliable protocol.
It is a best effort delivery service. i.e. there is no error checking in IP, it simply sends the data and relies on its underlying layers to get the data transmitted to the destination.Even if it is unreliable IP cannot be considered weak and useless; since it provides only the functionality that is requiredfor transmitting data thereby giving maximum efficiency.
Since there is no mechanism of error detection or correction in IP, there will be no delay introduced on a medium where thereis no error at all.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 17 | lines 3-15

The structure TCP/IP model is very similar to the structure of the OSI reference model. The OSI model has seven layers where the TCP/IP model has four layers.
• The Application layer of TCP/IP model corresponds to the Application Layer of Session, Presentation & Application Layer of OSI model.

• The Transport layer of TCP/IP model corresponds to theTransport Layer of OSI model
• The Network layer of TCP/IP model corresponds to the Network Layer of OSI model
• The Host to network layer of TCP/IP model correspondsto the Physical and Data link Layer of OSI model.

Question sources

Computer Science model exam collection · Question 52

Question 170 | SS170

Which of the following is the correct order of packet data unit as data moved in the OS! reference model from bottom to top?

A. Segment, packet, frame, bit

B. Segment, frame, packet, bit

C. Bit, packet, frame, segment

D. Bit, frame, packet, segment

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Bit, frame, packet, segment

Moving bottom to top reverses sending encapsulation: physical bits become data-link frames, then network packets, then transport segments. Evidence: PDF 14, 15, 15, 16.

Why each choice is right or wrong

A - Incorrect: Starts at transport instead of physical and moves downward

B - Incorrect: Starts at transport and misorders frame and packet

C - Incorrect: Skips frame before packet

D - Correct: Bit, frame, packet, segment follows the upward layer order

Primary note proof

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which

device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Question sources

Computer Science model exam collection · Question 77

Question 171 | AM0474

Which of the following is not a primary function of the network layer in the OSI model?

A. Routing

B. Addressing

C. Flow control

D. Fragmentation

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: Flow control is not a primary network-layer duty

The network layer supplies logical addressing/routing and, for IPv4, fragmentation when permitted. Flow control instead protects a receiver in transport or hop-local link procedures, as NET14 and NET17 explain. Thus C is not a primary network-layer function in this layer model, without claiming that all control mechanisms belong exclusively to one layer.

Why each choice is right or wrong

A - Incorrect: Routing is explicitly a main network-layer duty in the NET12 layer table.

B - Incorrect: Logical addressing is explicitly assigned to the network layer.

C - Correct: Receiver flow control is described at link and transport scope rather than as a primary network duty.

D - Incorrect: NET16 supports IPv4 fragmentation by sources and routers, a network-protocol function.

Primary note proof

Study-note section NET16

> The IPv4 checksum protects its header; IP does not lack error checking altogether. Identification, flags and offset support fragmentation. IPv4 sources and routers may fragment when allowed; reassembly occurs at the destination. If DF is set and the next MTU is too small, forwarding fails and an ICMP diagnostic may be returned.

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link

reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which

bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Question sources

Computer Science MCQ practice collection · Question 474

Question 172 | AM0475

What is the purpose of a MAC address in computer networking?

A. To identify a device within a local area network (LAN)

B. To identify a device within a wide area network (WAN)

C. To establish a secure connection between two devices

D. To determine the IP address of a device

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: To identify a device within a local area network (LAN)

The module physical MAC address is imprinted on a NIC and used for local physical delivery; it is not encryption or the IP routing address. Evidence: PDF 17, 15.

Why each choice is right or wrong

A - Correct: MAC identifies the local physical interface for frame delivery (PDF 17, 15.)

B - Incorrect: WAN-wide logical routing uses IP rather than treating MAC as a universal end-to-end address (PDF 17, 15.)

C - Incorrect: A physical identifier does not establish encryption/security (PDF 17, 15.)

D - Incorrect: ARP starts with IP and resolves MAC; MAC itself is not a rule for determining IP (PDF 17, 15.)

Primary note proof

Study-note section NET20

ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target's MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP

manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Study-note section NET10

A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to
resolve thephysical address of a device on a network,
where its logical address is known.
Physical address is the 48 bit address that is imprinted
on theNIC or LAN card, Logical address is the Internet
Address orcommonly known as IP address that is used
to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is
used bya device on the network to find its Internet
address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical
layer by providing error detection and correction
mechanisms.
• Framing: the Data Link layer receives the data
from Network Layer and divides the stream of bits
into fixed size manageable units called as Frames
and sends it to the physical layer. On the receiver
side, the data link layer receives the stream of bits
from the physical layer and regroups them into
frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends
the physical address in the header of the frame
before sending it to physical layer.

• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Question sources

Computer Science MCQ practice collection · Question 475

Question 173 | AM0476

Which of the following protocols is used for email transmission?

A. FTP (File Transfer Protocol)

B. SMTP (Simple Mail Transfer Protocol)

C. HTTP (Hypertext Transfer Protocol)

D. DNS (Domain Name System)

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer B: SMTP (Simple Mail Transfer Protocol)

SMTP is explicitly Simple Mail Transfer Protocol in the application list, matching email transmission. Evidence: PDF 18, 28.

Why each choice is right or wrong

A - Incorrect: FTP transfers files (PDF 18, 28.)

B - Correct: SMTP is the explicitly named mail transfer protocol (PDF 18, 28.)

C - Incorrect: HTTP is the web service in the port table (PDF 18, 28.)

D - Incorrect: DNS is external naming (PDF 18, 28.)

Primary note proof

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the
combined session, presentation, and application layers in
the OSI model and many protocols are defined at this
layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science MCQ practice collection · Question 476

Question 174 | AM0477

What is the purpose of a firewall in computer networks?

A. To encrypt network traffic for secure communication

B. To manage network resources and allocate bandwidth

C. To filter and control network traffic based on predefined rules

D. To connect two separate networks and enable communication between them

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: To filter and control network traffic based on predefined rules

The module firewall example permits only authorized systems to access internal computers, supporting selective traffic filtering/control. Evidence: PDF 48, 12, 16.

Why each choice is right or wrong

A - Incorrect: Encryption is a distinct presentation function (PDF 48, 12, 16.)

B - Incorrect: Bandwidth allocation is not the firewall access-control function described (PDF 48, 12, 16.)

C - Correct: Rules permitting authorized access match firewall filtering (PDF 48, 12, 16.)

D - Incorrect: Connecting networks is the router/gateway function, not the defining firewall policy (PDF 48, 12, 16.)

Primary note proof

Study-note section NET11

> A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

> Network Address Translation (NAT)
> NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing

authorized systems to access the computers within the network.

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device used tointerconnect different types of network that uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to forwardpackets.
• The list of routing tables provides directions to transfer the data to a particular network destination -various paths to forward the data.
• A router analyzes a destination IP address of a givenpacket and compares it with the routing table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Question sources

Computer Science MCQ practice collection · Question 477

Question 175 | AM0478

Which of the following network topologies provides the highest fault tolerance?

A. Bus topology

B. Ring topology

C. Star topology

D. Mesh topology

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer D: Mesh topology

Mesh has redundant interconnections and a single link failure does not affect the entire network. Bus backbone, ring break and star center failure are explicit vulnerabilities. Evidence: PDF 23, 24, 23, 23, 23.

Why each choice is right or wrong

A - Incorrect: A broken backbone can shut down bus (PDF 23, 24, 23, 23, 23.)

B - Incorrect: A break/disabled station can disable a ring (PDF 23, 24, 23, 23, 23.)

C - Incorrect: Central device failure can shut down star (PDF 23, 24, 23, 23, 23.)

D - Correct: Redundant mesh paths support the strongest listed fault tolerance (PDF 23, 24, 23, 23, 23.)

Primary note proof

Study-note section NET08

> Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Study-note section NET08

> A full undirected mesh of n nodes needs n(n-1)/2 links and n-1 interfaces per node. For 50 nodes: 50×49/2=1225 links. Redundant paths improve fault tolerance at greater cable and interface cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 23 | lines 45-53

> Mesh Topology: in this topology devices are
> connected with many redundant interconnections
> between network nodes. In a full mesh topology,
> every node has a connection to every other node in
> the network, which makes it the most expensive
> topology over all the other topologies. The number of
> cables grows fast as the number of nodes increases,
> and it can be calculated by using the general formula
> ((n (n - 1)) /2), where n is the number of nodes in the

Data Communication and Computer Networking module | PDF page 24 | lines 2-10

network. It has several benefits, such as: dedicated links between devices, robust (single link failure don‘t affect entire network), privacy/security (direct communication between communicating devices), and ease of fault identification and isolation. Its drawbacks include, installation and reconnection are difficult (large number of cables), huge amount of cables consumes a lot of space, and it is the most expensive of all.

Data Communication and Computer Networking module | PDF page 23 | lines 19-34

Star Topology: in this topology, each node is connected directly to a central device called a hub or a switch. Data on a star network passes through the central device (switch) before continuing to its destination. The central device manages and controls all functions of the network. This configuration is common with twisted pair cable. RJ-45 Connectors are used to connect the cable to the Network Interface Card (NIC) of each computer. Its advantages include, ease of installation and reconfiguration, robust (ease of fault identification and isolation), link failure only affects device(s) connected to that link, and is less expensive than mesh. Its drawbacks include more cable requirements (than bus and ring) and single point of failure (if central device fail, the whole system will be down).

Data Communication and Computer Networking module | PDF page 23 | lines 35-44

Ring Topology: in this topology, all devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it. Some of its advantages include, easy to install and reconfigure, less expensive (than mesh), and performance is even despite the number of users. Its cons include, break in the ring (such as a disabled station) can disable the entire network, and limitations on media and traffic (limitation on ring length and number of devices).

Data Communication and Computer Networking module | PDF page 23 | lines 4-18

Bus Topology: in this topology all devices are connected to a central cable, called the bus or backbone, which is terminated at its ends. The purpose of the terminators is to stop the signal from bouncing, thereby clearing the cable so that other computers can send data. Message transmitted along

the Bus is visible to all computers connected to the backbone cable. As the message arrives at each workstation, the workstation checks the destination address contained in the message either to process or drop the packet if it matches or not respectively. Its advantages are, ease of installation and less amount of cable requirement. Its main drawback is, the entire network will be shut down if there is a break in the main cable.

Question sources

Computer Science MCQ practice collection · Question 478

Question 176 | AM0479

Which listed terms describe IPv6’s flexible-prefix addressing or operations supported by that approach? Select all that apply.

A. Classful addressing

B. Classless addressing

C. Subnetting

D. CIDR (Classless Inter-Domain Routing)

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer B, C, D: Classless addressing; Subnetting; CIDR (Classless Inter-Domain Routing)

IPv6 uses flexible network prefixes rather than historical IPv4 address classes. Classless addressing describes this approach, subnetting divides prefixes into smaller networks, and CIDR terminology expresses classless prefixes and aggregation. These are compatible descriptions or operations rather than mutually exclusive schemes. The selection instruction therefore permits all three while excluding the classful A/B/C allocation model associated with historical IPv4.

Why each choice is right or wrong

A - Incorrect: Historical classful A/B/C IPv4 defaults are not the IPv6 flexible-prefix scheme.

B - Correct: IPv6 uses flexible prefixes rather than fixed historical address classes.

C - Correct: IPv6 prefixes can be subdivided into subnets; the term overlaps the broad scheme wording.

D - Correct: Classless prefix/CIDR terminology applies to flexible-prefix allocation and routing rather than historical class boundaries.

Primary note proof

Study-note section NET23

IPv6 has 128-bit hexadecimal addresses, flexible prefixes and no IPv4-style broadcast. Sources, not routers, fragment IPv6 packets. IPv6 does not automatically encrypt traffic or make authentication unnecessary. Compare IPv6's larger 128-bit address space and encryption support separately; specify the property being evaluated.

Study-note section NET28

CIDR, Classless Inter-Domain Routing, replaces rigid class allocation and supports aggregation. Combine contiguous aligned blocks sharing leading bits; unrelated adjacent-looking labels may not aggregate cleanly. Four consecutive /26 blocks covering a /24 summarize to /24. More-specific routes must remain available where the aggregate would otherwise send traffic to the wrong next hop.

Study-note section NET26

To split parent /a into equal /p blocks, borrow b=p-a bits and obtain 2^b subnets. Do not subtract two subnets under modern practice. For a Class B /16 parent split to /19, b=3 gives eight subnets, each with 8192 addresses and 8190 conventional hosts. A /19 alone does not state its parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 37-44

CIDR: A class B address is far too large for most
organizationsand a class C network, with 256 addresses is
too small. This leads to granting Class B address to
organizations who do not require all. This is resulting in
depletion of Address space. A solution is CIDR (Classless
InterDomain Routing) The basic idea behind CIDR, is to
allocate the remaining IP addresses in variable-sized
blocks, without regard to the classes.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of
network addresses in IPv4 led to the development of IPv6.
Due to this, we need to use private IP address inside the
organization and translate it to public IP address using
NAT.

Question sources

Computer Science MCQ practice collection · Question 479

Question 177 | AM0480

Which protocol is responsible for translating domain names into IP addresses?

A. DHCP (Dynamic Host Configuration Protocol)

B. DNS (Domain Name System)

C. ARP (Address Resolution Protocol)

D. ICMP (Internet Control Message Protocol)

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer B: DNS (Domain Name System)

DNS is the domain naming service and external name/address service contrasted with hosts entries. Evidence: PDF 18, 65, 28, 17, 17.

Why each choice is right or wrong

A - Incorrect: DHCP is named host configuration, not domain naming

B - Correct: DNS matches the named domain service

C - Incorrect: ARP resolves known IP to MAC

D - Incorrect: ICMP reports packet problems

Primary note proof

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET20

> ARP resolves an on-link IPv4 next hop to a link address, using a broadcast request and normally unicast reply. For an off-subnet target, ARP resolves the gateway, not the remote target's MAC. It bridges the network/link addressing boundary and is often classified at the link layer; calling ARP simply a network protocol uses a loose textbook convention. RARP reverses hardware-to-IP lookup; BOOTP/DHCP support boot configuration. ICMP reports errors and diagnostics; IGMP manages IPv4 multicast membership. These are separate companion protocols, not four pieces whose combination is IP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

/etc/hosts
The /etc/hosts file always contains the localhost IP address, 127.0.0.1, which is
usedfor interprocess communication.
Never remove this line! Sometimes contains addresses of additional hosts, whichcan be contacted without using an external naming service such as DNS (the Domain Name Server).

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the combined session, presentation, and application layers in the OSI model and many protocols are defined at this layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 17 | lines 44-53

ARP - Address Resolution Protocol: It is used to resolve thephysical address of a device on a network, where its logical address is known.
Physical address is the 48 bit address that is imprinted on theNIC or LAN card, Logical address is the Internet Address orcommonly known as IP address that is used to uniquely & universally identify a device.
RARP- Reverse Address Resolution Protocol: It is used bya device on the network to find its Internet address when it knows its physical address.

Data Communication and Computer Networking module | PDF page 17 | lines 54-59

ICMP- Internet Control Message Protocol: It is a signaling mechanism used to inform the sender about datagram problemsthat occur during transit. It is used by intermediate devices.
IGMP- Internet Group Message Protocol: It is a mechanism that allows sending the same message to a group of recipients.

Question sources

Computer Science MCQ practice collection · Question 480

Question 178 | AM0481

What is the purpose of NAT (Network Address Translation) in computer networks?

A. To secure network communications through encryption

B. To provide redundancy and fault tolerance

C. To translate private IP addresses to public IP addresses

D. To manage network resources and allocate bandwidth

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer C: To translate private IP addresses to public IP addresses

NAT explicitly translates an organization private IP address to a public IP address. It is not an encryption, redundancy or bandwidth-allocation mechanism. Evidence: PDF 19, 48, 16.

Why each choice is right or wrong

A - Incorrect: Encryption is a separate representation/security transformation (PDF 19, 48, 16.)

B - Incorrect: Redundancy is not the address translation definition (PDF 19, 48, 16.)

C - Correct: Private-to-public translation is the exact NAT definition (PDF 19, 48, 16.)

D - Incorrect: Bandwidth allocation is not the NAT definition (PDF 19, 48, 16.)

Primary note proof

Study-note section NET29

Static NAT gives a fixed address mapping; dynamic NAT allocates mappings from a pool. PAT/NAPT maps address-plus-port combinations so multiple private clients share a public address. It is translation, not ARP resolution and not encryption. Return traffic must match the translation state or a configured inbound mapping.

Study-note section NET29

A private host such as 10.0.0.1 commonly reaches the public Internet through NAT at a gateway, together with valid routing and a public external address. NAT alone does not supply a physical

connection or DNS. A public address does not automatically grant permission to use it; both public and private unicast addresses can be used on a LAN under the correct allocation and subnet.

Study-note section NET11

A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Related reading (context, not primary note proof)

Study-note section NET20

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of network addresses in IPv4 led to the development of IPv6. Due to this, we need to use private IP address inside the organization and translate it to public IP address using NAT.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.

• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Variant 1 | Alternative version 1

What is the purpose of NAT (Network Address Translation) in computer networks?

A. To encrypt network traffic for secure communication

B. To assign unique IP addresses to hosts in a network

C. To provide a firewall for network security

D. To translate private IP addresses to public IP addresses

Answer D: To translate private IP addresses to public IP addresses

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Incorrect: Encryption changes data protection; NAT changes addresses (PDF19,48).

B - Incorrect: Assigning initial unique host addresses is different from translating already-used internal addresses (PDF19,48).

C - Incorrect: PDF48 says a router can additionally be a firewall; optional access control is not the NAT translation definition.

D - Correct: This variant moves the explicit private-to-public translation definition to D (PDF19).

Primary note proof

Study-note section NET29

Static NAT gives a fixed address mapping; dynamic NAT allocates mappings from a pool. PAT/NAPT maps address-plus-port combinations so multiple private clients share a public address. It is translation, not ARP resolution and not encryption. Return traffic must match the translation state or a configured inbound mapping.

Study-note section NET20

DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET11

A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Question sources

Computer Science MCQ practice collection · Question 481

Computer Science MCQ practice collection · Question 522

Question 179 | AM0482

Which listed protocol is a connectionless transport-layer protocol, sending independent datagrams without built-in connection setup?

A. TCP (Transmission Control Protocol)

B. UDP (User Datagram Protocol)

C. HTTP (Hypertext Transfer Protocol)

D. FTP (File Transfer Protocol)

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer B: UDP (User Datagram Protocol)

UDP is a transport protocol that sends independent datagrams without a built-in setup handshake, ordering or retransmission service. TCP instead establishes a connection and supplies an ordered reliable byte stream. HTTP and FTP are application protocols, so they are outside the requested transport-layer category. HTTP being stateless at the application level does not make it a connectionless transport protocol.

Why each choice is right or wrong

A - Incorrect: TCP is connection-oriented transport with reliable ordered byte-stream delivery.

B - Correct: UDP is the connectionless datagram transport described.

C - Incorrect: HTTP is an application protocol; application statelessness is distinct from connectionless transport.

D - Incorrect: FTP is an application file-transfer protocol, not a transport protocol.

Primary note proof

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Computer Science MCQ practice collection · Question 482

Question 180 | AM0483

Which listed devices ordinarily forward frames at OSI layer 2? Select all that apply.

A. Hub

B. Switch

C. Router

D. Bridge

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B, D: Switch; Bridge

A bridge and an ordinary Ethernet switch both inspect link-layer frames and use MAC-address information in forwarding. A hub regenerates and repeats physical signals without that address inspection. A router forwards IP packets between networks using network-layer destinations. The explicit multiple-selection instruction preserves both correct layer-2 devices and avoids implying that only one kind of device can operate at a layer.

Why each choice is right or wrong

A - Incorrect: A hub repeats layer-1 signals rather than making MAC/frame forwarding decisions.

B - Correct: An ordinary Ethernet switch is explicitly a layer-2 forwarding device.

C - Incorrect: A router's defining forwarding function is network-layer IP routing.

D - Correct: A bridge is explicitly a layer-2 device, like an ordinary Ethernet switch.

Primary note proof

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET09

> A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

> SWITCH: A switch filters and forwards data packets
> across anetwork.

Unlike a hub which simply replicates what it receives on oneport onto all the other ports, a switch keeps a record of the MAC addresses of the devices attached to it. When the switch receives a data packet, it forwards the packetdirectly to the recipient device by looking up the MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 12 | lines 60-61

BRIDGE: Connects two LANS having the same protocol - (e.g. Ethernet or Token ring).

Data Communication and Computer Networking module | PDF page 13 | lines 2-8

Filters content by reading the MAC addresses of source and destination.
• MAC to Segment number table
• Do not filter broadcast
• Data not destined for other network is prevented from passing over the bridge
• Slower than repeater due to filtering

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes

access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is normallyused to create connections between stations in a physical startopology.
HUB: Hubs are commonly used to connect segments (devices)of a LAN. Hubs don‘t filter information; instead broadcast incoming packets to all computers connected to it. A hub contains multiple ports which support 8, 12 or 24 RJ-45 ports. Two types: Passive hub and Active hub. It is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device used tointerconnect different types of network that uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to forwardpackets.
• The list of routing tables provides directions to transfer the data to a particular network destination -various paths to forward the data.
• A router analyzes a destination IP address of a givenpacket and compares it with the routing table to decide the packet's next best path

Question sources

Computer Science MCQ practice collection · Question 483

Question 181 | AM0485

What is the function of a router in a computer network?

A. To connect multiple networks together

B. To amplify the strength of network signals

C. To provide firewall protection

D. To manage network resources and allocate bandwidth

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer A: To connect multiple networks together

The router definition explicitly interconnects LANs/WANs and determines next paths using routing tables. Firewall functionality may be added, but it is not the primary definition. Evidence: PDF 12, 11, 48.

Why each choice is right or wrong

A - Correct: Connecting networks is the explicit router role (PDF 12, 11, 48.)

B - Incorrect: Signal regeneration is repeater work (PDF 12, 11, 48.)

C - Incorrect: A router can have firewall configuration, but this is optional additional policy (PDF 12, 11, 48.)

D - Incorrect: Bandwidth allocation is not the defining routing role (PDF 12, 11, 48.)

Primary note proof

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Study-note section NET11

> A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Study-note section NET09

> A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor

eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device
used tointerconnect different types of network that
uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to
forwardpackets.
• The list of routing tables provides directions to
transfer the data to a particular network
destination -various paths to forward the data.
• A router analyzes a destination IP address of a
givenpacket and compares it with the routing
table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 11 | lines 54-55

Repeater: A repeater is a device that operates only in the physical
layer. A repeater receives a signal and, before it

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local
network to a single IP address. This address is often used
by the router that connects the computers to the Internet.
The router can be connected to a DSL modem, cable
modem, T1 line, or even a dial-up modem. When other
computers on the Internet attempt to access computers
within the local network, they only see the IP address of
the router. This adds an extra level of security, since the
router can be configured as a firewall, only allowing
authorized systems to access the computers within the
network.

Question sources

Computer Science MCQ practice collection · Question 485

Question 182 | AM0486

Which of the following is a network device that operates at the Physical layer of the OSI model?

A. Switch

B. Hub

C. Router

D. Bridge

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Hub

An active hub is explicitly a multiport repeater, and repeaters operate at physical layer. Evidence: PDF 12, 11, 12, 12, 12.

Why each choice is right or wrong

A - Incorrect: Switch forwards by MAC at data-link scope (PDF 12, 11, 12, 12, 12.)

B - Correct: Active hub repeats physical signals (PDF 12, 11, 12, 12, 12.)

C - Incorrect: Router selects IP paths (PDF 12, 11, 12, 12, 12.)

D - Incorrect: Bridge filters by MAC (PDF 12, 11, 12, 12, 12.)

Primary note proof

Study-note section NET09

> A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

Active Hubs
An active hub is actually a multipart repeater. It is normallyused to create connections between stations in a physical startopology.
HUB: Hubs are commonly used to connect segments (devices)of a LAN. Hubs don‘t filter information; instead broadcast incoming packets to all computers connected to it. A hub contains multiple ports which support 8, 12 or 24 RJ-45 ports. Two types: Passive hub and Active hub. It is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 11 | lines 54-55

Repeater: A repeater is a device that operates only in the physical layer. A repeater receives a signal and, before it

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets across anetwork.
Unlike a hub which simply replicates what it receives on oneport onto all the other ports, a switch keeps a record of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the packetdirectly to the recipient device by looking up the MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 12 | lines 38-52

ROUTER: Router is a specialized network device used tointerconnect different types of network that uses different protocol.
Router allows the users to connect several LAN and WAN
• A LAN to another LAN
• A WAN to another WAN
• A LAN to the Internet
Routers use the routing table to determine how to forwardpackets.
• The list of routing tables provides directions to transfer the data to a particular network destination -various paths to forward the data.
• A router analyzes a destination IP address of a

givenpacket and compares it with the routing
table to decide the packet's next best path

Data Communication and Computer Networking module | PDF page 12 | lines 60-61

BRIDGE: Connects two LANS having the same protocol -
(e.g. Ethernet or Token ring).

Question sources

Computer Science MCQ practice collection · Question 486

Question 183 | AM0487

Which network component is responsible for converting digital data into analog signals for transmission over telephone lines?

A. Modem

B. Router

C. Switch

D. Firewall

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer A: Modem

The stem asks about conversion for a telephone-line carrier, which is modem modulation. NET11 separates this signal-conversion function from routing IP packets, switching link frames and applying firewall traffic policy. These functions can coexist in one appliance, but the functional component asked for remains the modem.

Why each choice is right or wrong

A - Correct: A modem modulates outgoing carrier signals and demodulates incoming ones, supplying the requested conversion.

B - Incorrect: A router selects next hops between networks; that role is distinct from telephone-carrier modulation.

C - Incorrect: A switch forwards link-layer frames and is not the component defined by carrier modulation.

D - Incorrect: A firewall applies security policy to traffic rather than converting data onto a telephone-line carrier.

Primary note proof

Study-note section NET11

A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Study-note section NET10

A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 13 | lines 65-77

• Digital modulation is used to convert digital data to analog signal. Ex ASK, FSK, PSK.
Analog Modulation can be accomplished in three ways:
• Amplitude modulation (AM)
• Frequency modulation (FM)
• Phase modulation (PM).
Digital Modulation Types (Digital to Analog signal conversion)
Digital modulation is used to convert digital data to analog signal. It can be accomplished in the following ways:
• Amplitude Shift Keying(ASK)
• Frequency Shift keying (FSK)
• Phase Shift keying (PSK)

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data

transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Question sources

Computer Science MCQ practice collection · Question 487

Question 184 | AM0488

Which protocol submits and relays email messages, including acceptance by a receiving mail server? Select one.

A. SMTP

B. HTTP

C. FTP

D. DNS

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A: SMTP

SMTP is used for mail submission and relay, including a mail server accepting an incoming message. End-user access to stored mail is a separate operation supplied by POP3 or IMAP. HTTP handles web exchanges, FTP transfers files, and DNS resolves records. Specifying server receipt and relay makes the mail-system role clear without assigning mailbox retrieval to the transfer protocol.

Why each choice is right or wrong

A - Correct: SMTP submits and relays mail, including server acceptance of incoming messages.

B - Incorrect: HTTP defines web request/response exchanges rather than the mail submission and relay protocol requested.

C - Incorrect: FTP transfers files and is not the mail submission/relay or mailbox access protocol in the note.

D - Incorrect: DNS resolves records; it does not transport or retrieve the user’s messages.

Primary note proof

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and

IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET20

DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the combined session, presentation, and application layers in the OSI model and many protocols are defined at this layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Computer Science MCQ practice collection · Question 488

Question 185 | AM0489

What is the purpose of DNS (Domain Name System) in computer networks?

A. To encrypt network traffic for secure communication

B. To manage network resources and allocate bandwidth

C. To convert domain names into IP addresses

D. To provide firewall protection

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer C: To convert domain names into IP addresses

The domain naming service is contrasted with explicit host-address entries, supporting resolving a name into the corresponding address. Evidence: PDF 18, 65.

Why each choice is right or wrong

A - Incorrect: Encryption is not DNS naming (PDF 18, 65.)

B - Incorrect: Bandwidth allocation is not DNS naming (PDF 18, 65.)

C - Correct: Name-to-address resolution matches Domain Name Service and the hosts-file contrast (PDF 18, 65.)

D - Incorrect: Firewall access policy is not DNS naming (PDF 18, 65.)

Primary note proof

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET11

> A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Study-note section NET-S02

> Quality of service (QoS) controls traffic treatment to meet objectives such as throughput, delay, jitter or loss. Classification, queue scheduling and allocation of buffer or bandwidth resources implement such treatment. QoS concerns delivered service behavior; scalability concerns accommodating growth, and security concerns unauthorized actions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

> Application Layer
> The Application Layer is a combination of Session,
> Presentation & Application Layers of OSI models and defines
> high level protocols like File Transfer (FTP), Electronic Mail
> (SMTP), Virtual Terminal (TELNET), Domain Name Service
> (DNS), etc.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

> /etc/hosts
> The /etc/hosts file always contains the localhost IP
> address, 127.0.0.1, which is
> usedfor interprocess communication.
> Never remove this line! Sometimes contains
> addresses of additional hosts, whichcan be contacted
> without using an external naming service such as
> DNS (the Domain Name Server).

Question sources

Computer Science MCQ practice collection · Question 489

Question 186 | AM0490

Which listed application protocol has the name File Transfer Protocol?

A. HTTP

B. FTP

C. SMTP

D. TCP

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer B: FTP

FTP is the named application protocol File Transfer Protocol. HTTP also transfers resource bodies and can download files, so generic file transfer alone does not distinguish them. SMTP is a mail-transfer application protocol, while TCP supplies an underlying reliable byte-stream transport. The explicit protocol name identifies the intended service without claiming that FTP is the only way to transfer a file.

Why each choice is right or wrong

A - Incorrect: HTTP is Hypertext Transfer Protocol and can carry files, but is not the stated protocol name.

B - Correct: FTP expands to File Transfer Protocol.

C - Incorrect: SMTP sends and relays email rather than bearing the stated protocol name.

D - Incorrect: TCP is a transport protocol, not the named application protocol.

Primary note proof

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

> Application Layer
> The Application Layer is a combination of Session,
> Presentation & Application Layers of OSI models and defines
> high level protocols like File Transfer (FTP), Electronic Mail
> (SMTP), Virtual Terminal (TELNET), Domain Name Service
> (DNS), etc.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

> Application Layer
> The application layer in TCP/IP is equivalent to the
> combined session, presentation, and application layers in
> the OSI model and many protocols are defined at this
> layer.
> The Process/Application Layer Protocols
> • Telnet
> • File Transfer Protocol (FTP)
> • Network File System (NFS).
> • Simple Mail Transfer Protocol (SMTP
> • POP3 is used to receive mail.

• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Question sources

Computer Science MCQ practice collection · Question 490

Question 187 | AM0491

Which device connects multiple Ethernet LAN endpoints and learns MAC-to-port mappings to forward known unicast frames selectively?

A. Hub

B. Router

C. Switch

D. Modem

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer C: Switch

Both hubs and switches can connect LAN endpoints, but their forwarding behavior differs. A hub repeats incoming signals to its other ports without learning addresses. An ordinary Ethernet switch learns source MAC-to-port mappings and uses the destination MAC address to forward known unicasts. Routers instead forward IP packets between networks, while modems convert carrier signals. The added forwarding requirement distinguishes the intended device.

Why each choice is right or wrong

A - Incorrect: A hub connects endpoints but repeats signals without learning MAC-to-port mappings.

B - Incorrect: A router's defining function is connecting/routing between IP networks rather than this shared LAN-device role.

C - Correct: A switch learns MAC-to-port mappings and forwards known unicasts selectively.

D - Incorrect: A modem modulates/demodulates carrier signals rather than providing the offered multiport LAN interconnection function.

Primary note proof

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET09

> A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Study-note section NET11

> A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 12 | lines 17-26

> Active Hubs
> An active hub is actually a multipart repeater. It is normallyused to create connections between stations in a physical startopology.
> HUB: Hubs are commonly used to connect segments (devices)of a LAN. Hubs don‘t filter information; instead broadcast incoming packets to all computers connected to it. A hub contains multiple ports which support 8, 12 or 24 RJ-45 ports. Two types: Passive hub and Active hub. It is used in star or ring topology.

Data Communication and Computer Networking module | PDF page 12 | lines 27-37

SWITCH: A switch filters and forwards data packets across anetwork.
Unlike a hub which simply replicates what it receives on oneport onto all the other ports, a switch keeps a record of the MAC addresses of the devices attached to it.
When the switch receives a data packet, it forwards the packetdirectly to the recipient device by looking up the MAC address.
• Switch is intelligent and expensive than Hub
• Switch is a point to point communication device
• Hub is a broadcasting device

Data Communication and Computer Networking module | PDF page 22 | lines 26-44

Personal Area Network (PAN): is a network may be spanned across a given table with distances between the devices not more than few meters. The technology used to interconnect the devices could be a Bluetooth. These networks are called Personal Area Networks, since the devices interconnected in these networks are belongs to a single person.
Local Area Network (LAN): is a network that may span across a building, or across several buildings within a single organization, by using intermediate devices, like switches and/or hubs, to interconnect devices in all floors.
Metropolitan Area Network (MAN): is a network that may span across a whole city interconnecting several buildings and organizations.
Wide Area Network (WAN): is a network that may span across multiple cities, or an entire country, or an entire continent or it may even cover the whole world. For example, an Internet is one example of WAN.

Question sources

Computer Science MCQ practice collection · Question 491

Question 188 | AM0492

What is the primary function of a firewall in a computer network?

A. To manage network resources and allocate bandwidth

B. To encrypt network traffic for secure communication

C. To filter and monitor network traffic for security purposes

D. To convert domain names into IP addresses

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: To filter and monitor network traffic for security purposes

The firewall example controls which systems may access internal computers; monitoring/filtering for security is the offered match. Evidence: PDF 48, 16, 18.

Why each choice is right or wrong

A - Incorrect: Resource/bandwidth allocation is not the described firewall role (PDF 48, 16, 18.)

B - Incorrect: Encryption is a presentation function and distinct from access filtering (PDF 48, 16, 18.)

C - Correct: Authorized-access filtering is the module firewall function (PDF 48, 16, 18.)

D - Incorrect: DNS is the naming service, not firewall policy (PDF 48, 16, 18.)

Primary note proof

Study-note section NET11

> A firewall allows, rejects or inspects traffic under a security policy; it does not necessarily encrypt. A VPN, Virtual Private Network, provides a logical private connection over another network; a secure VPN authenticates/encrypts its tunnel. A modem modulates outgoing and demodulates incoming carrier signals. These functions may coexist in one home router but remain conceptually distinct.

Study-note section NET-S02

> Quality of service (QoS) controls traffic treatment to meet objectives such as throughput, delay, jitter or loss. Classification, queue scheduling and allocation of buffer or bandwidth resources implement such treatment. QoS concerns delivered service behavior; scalability concerns accommodating growth, and security concerns unauthorized actions.

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

> Network Address Translation (NAT)
> NAT translates the IP addresses of computers in a local
> network to a single IP address. This address is often used
> by the router that connects the computers to the Internet.
> The router can be connected to a DSL modem, cable

modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Computer Science MCQ practice collection · Question 492

Question 189 | AM0493

Which protocol is responsible for assigning IP addresses to devices on a network automatically?

A. HTTP

B. DNS

C. DHCP

D. FTP

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer C: DHCP

NET20 assigns dynamic address-setting leases to DHCP, whose basic exchange is discover, offer, request and acknowledge. DNS instead resolves names to records. HTTP and FTP are application services for web exchanges and file transfer; neither is the automatic host-address lease protocol.

Why each choice is right or wrong

A - Incorrect: HTTP provides web request/response semantics rather than automatic IP-address leases.

B - Incorrect: DNS resolves names into records, including addresses, but does not lease the host's own address.

C - Correct: DHCP dynamically leases an IP address and related host-network configuration.

D - Incorrect: FTP transfers files; it does not provide the address-assignment exchange described.

Primary note proof

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

> Application Layer
> The application layer in TCP/IP is equivalent to the
> combined session, presentation, and application layers in
> the OSI model and many protocols are defined at this
> layer.
> The Process/Application Layer Protocols
> • Telnet
> • File Transfer Protocol (FTP)
> • Network File System (NFS).
> • Simple Mail Transfer Protocol (SMTP
> • POP3 is used to receive mail.
> • Simple Network Management Protocol (SNMP)

• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science MCQ practice collection · Question 493

Question 190 | AM0494

Which OSI layer establishes and manages application dialogs, including checkpoints for recovery after an interruption?

A. Physical layer

B. Data Link layer

C. Transport layer

D. Session layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Session layer

Session-layer roles include establishing and managing dialogs and coordinating checkpoints so a transfer can resume after an interruption. TCP transport connections are a different form of connection state and do not make every reference to a dialog a transport function. Physical signaling and hop-local link delivery likewise have different scopes. The revised stem identifies dialog and checkpoint management as the lifecycle being classified.

Why each choice is right or wrong

A - Incorrect: Physical signaling does not perform application-dialog management or checkpoint coordination.

B - Incorrect: Data link handles hop-local frames and link procedures rather than application dialogs and checkpoints.

C - Incorrect: Transport handles process delivery and TCP connection state, not the session-dialog role specified.

D - Correct: Session manages dialogs and can coordinate checkpoints for recovery.

Primary note proof

Study-note section NET19

OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations

distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET18

TCP establishes state with SYN, SYN+ACK, then ACK. SYN consumes one sequence position even with no application bytes. An acknowledgment states the next expected sequence number. For client ISN=395 and server ISN=260: client sends SYN seq395; server sends SYN+ACK seq260 ack396; client sends ACK seq396 ack261. Later byte counts advance sequence numbers. FIN also consumes one sequence position.

Study-note section NET14

Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 18-32

Session Layer
• The session layer establishes a session between the communicating devices called dialog and synchronizes their interaction. It is the responsibility of the session layer to establish and synchronize the dialogs. It is also called the network dialog controller.
• The session layer at the sending side accepts data from the presentation layer adds checkpoints to it called sync bits and passes the data to the transport layer. At the receiving end the session layer receives data from the transport layer removes the checkpoints inserted previously and passes the data to the presentation layer. So, the main responsibility of session layer is dialog control and synchronization.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.

• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the

receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Question sources

Computer Science MCQ practice collection · Question 494

Question 191 | AM0495

Which protocol is used for reliable data delivery at the transport layer in the TCP/IP protocol suite?

A. UDP

B. IP

C. FTP

D. TCP

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: TCP

TCP is explicitly reliable and connection-oriented at transport, unlike UDP and network-layer IP. Evidence: PDF 18, 17, 18.

Why each choice is right or wrong

A - Incorrect: UDP is explicitly unreliable (PDF 18, 17, 18.)

B - Incorrect: IP is network-layer best effort (PDF 18, 17, 18.)

C - Incorrect: FTP is application file transfer (PDF 18, 17, 18.)

D - Correct: TCP is the explicitly reliable transport protocol (PDF 18, 17, 18.)

Primary note proof

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable connection-oriented, reliable protocol. i.e. a connection is established between the sender and receiver before the data can be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 17 | lines 22-38

Network Layer or IP
Also called as the Internetwork Layer (IP). It holds the IP protocol which is a network layer protocol and is responsiblefor source to destination transmission of data.
The Internetworking Protocol (IP) is a connection-less & unreliable protocol.
It is a best effort delivery service. i.e. there is no error checking in IP, it simply sends the data and relies on its

underlying layers to get the data transmitted to the destination.Even if it is unreliable IP cannot be considered weak and useless; since it provides only the functionality that is requiredfor transmitting data thereby giving maximum efficiency.
Since there is no mechanism of error detection or correction in IP, there will be no delay introduced on a medium where thereis no error at all.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Computer Science MCQ practice collection · Question 495

Question 192 | AM0496

Which layer of the OSI reference model is responsible for addressing and routing data packets across different networks?

A. Network layer

B. Data Link layer

C. Transport layer

D. Physical layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: Network layer

Network explicitly adds logical addressing and routes packets from source to destination across intermediate devices. Evidence: PDF 15, 15, 16, 14.

Why each choice is right or wrong

A - Correct: Network has addressing and routing responsibilities (PDF 15, 15, 16, 14.)

B - Incorrect: Data link handles physical addresses and one-hop frames (PDF 15, 15, 16, 14.)

C - Incorrect: Transport delivers to processes with ports (PDF 15, 15, 16, 14.)

D - Incorrect: Physical delivers bits/signals (PDF 15, 15, 16, 14.)

Primary note proof

Study-note section NET30

Routing computes paths; forwarding chooses a next hop for an individual packet. Select the matching route with the longest prefix, then apply the routing system's tie-breaking criteria. A default /0 matches when nothing more specific does. Static routes are configured; dynamic protocols exchange reachability. Distance vector uses neighbor estimates, while link state distributes link information and computes shortest paths. Interior protocols operate within an autonomous system; BGP exchanges routes between systems.

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Study-note section NET17

Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control

• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and

error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Question sources

Computer Science MCQ practice collection · Question 496

Question 193 | AM0497

Which OSI layer supplies hop-local frame acknowledgments and retransmission when a reliable-link protocol is used, rather than end-to-end transport recovery?

A. Network layer

B. Transport layer

C. Data Link layer

D. Application layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: Data Link layer

Reliable data-link protocols can acknowledge frames and retransmit missing or damaged frames across a single hop. Transport recovery instead covers the end-to-end communication between processes. Network-layer IPv4 includes a header checksum but remains best-effort and does not supply this frame ARQ service. The link-layer role is protocol-dependent: Ethernet error detection alone does not guarantee reliable local delivery.

Why each choice is right or wrong

A - Incorrect: IP routing and its header checksum are not the requested hop-local frame ARQ service.

B - Incorrect: Transport reliability has end-to-end rather than one-link scope.

C - Correct: Data link implements the specified hop-local frame acknowledgment/retransmission procedure.

D - Incorrect: Application services are above the specified link-frame recovery mechanism.

Primary note proof

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Study-note section NET13

> Detection adds redundancy but does not guarantee correction. Even parity adds a bit so the total number of ones is even: 1011 needs parity 1. It detects every odd number of bit flips but misses even counts. CRC treats bits as a polynomial and appends the XOR-division remainder; the receiver repeats the division. Internet checksums add words with end-around carry and complement the sum. A burst spans from the first corrupted bit through the last, including unchanged interior bits. Retransmission or sufficient forward-error-correcting redundancy is needed for recovery.

Study-note section NET16

> The IPv4 checksum protects its header; IP does not lack error checking altogether. Identification, flags and offset support fragmentation. IPv4 sources and routers may fragment when allowed; reassembly occurs at the destination. If DF is set and the next MTU is too small, forwarding fails and an ICMP diagnostic may be returned.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 15 | lines 13-40

Data Link Layer
• The Data Link layer adds reliability to the physical layer by providing error detection and correction mechanisms.
• Framing: the Data Link layer receives the data from Network Layer and divides the stream of bits into fixed size manageable units called as Frames and sends it to the physical layer. On the receiver side, the data link layer receives the stream of bits from the physical layer and regroups them into frames and sends them to the Network layer.
• Physical Addressing: The Data link layer appends the physical address in the header of the frame before sending it to physical layer.
• Flow control: The data link layer makes sure that the sender sends the data at a speed at which the receiver canreceive it else if there is an overflow at the receiver sidethe data will be lost.
• Error Control: The data link layer imposes error control mechanism to identify lost or damaged frames, duplicateframes and then retransmit them.
• Access Control: The data link layer imposes access control mechanism to determine which device has right to send data in an multipoint connection setting.
• Main Responsibility: The main responsibility of the data link layer is hop to hop transmission of frames.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 15 | lines 41-61

Network Layer
• The network layer makes sure that the data is deliveredto the receiver despite multiple intermediate devices.
• The network layer at the sending side accepts data from the transport layer, divides it into packets, addsaddressing information in the header and passes it tothe data link layer.
• The network layer is responsible for source todestination of delivery of data.
• The network layer uses logical address commonlyknown as IP address to recognize devices on the network.
• Each packet is independent of the other and may travelusing different routes to reach the receiver hence may arrive out of turn at the receiver (Routing).
• The Network layer does not perform any flow controlor error control
• The main responsibility of Network Layer is transmission of packets from source to destination

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Question sources

Computer Science MCQ practice collection · Question 497

Question 194 | AM0498

Which protocol submits and relays email messages, including acceptance by a receiving mail server? Select one.

A. SMTP

B. FTP

C. HTTP

D. DNS

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A: SMTP

SMTP is used for mail submission and relay, including a mail server accepting an incoming message. End-user access to stored mail is a separate operation supplied by POP3 or IMAP. HTTP handles web exchanges, FTP transfers files, and DNS resolves records. Specifying server receipt and relay makes the mail-system role clear without assigning mailbox retrieval to the transfer protocol.

Why each choice is right or wrong

A - Correct: SMTP submits and relays mail, including server acceptance of incoming messages.

B - Incorrect: FTP transfers files through a separate protocol rather than the mail flow defined here.

C - Incorrect: HTTP defines web request/response exchanges rather than the mail submission and relay protocol requested.

D - Incorrect: DNS supplies records such as mail-exchanger names rather than delivering or retrieving message contents.

Primary note proof

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

> Application Layer
> The application layer in TCP/IP is equivalent to the
> combined session, presentation, and application layers in
> the OSI model and many protocols are defined at this
> layer.
> The Process/Application Layer Protocols
> • Telnet
> • File Transfer Protocol (FTP)
> • Network File System (NFS).
> • Simple Mail Transfer Protocol (SMTP

• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Question sources

Computer Science MCQ practice collection · Question 498

Question 195 | AM0499

Which layer of the OSI reference model is responsible for converting data into a format suitable for transmission over a network?

A. Presentation layer

B. Session layer

C. Physical layer

D. Application layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: Presentation layer for representation/format translation

The word format identifies representation translation rather than routing or connection management. NET19 assigns encoding translation and other representation work to presentation. Physical signaling maps bits to medium symbols, but does not supply the application-data format translation asked for. Thus A matches the intended formatting responsibility.

Why each choice is right or wrong

A - Correct: Presentation translates representations/encodings into suitable data formats.

B - Incorrect: Session manages dialogs and checkpoints rather than data representation formats.

C - Incorrect: Physical maps bits/symbols to signals, a different responsibility from representation-format translation.

D - Incorrect: Application supplies network services to applications rather than the distinct presentation-format function.

Primary note proof

Study-note section NET19

OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET09

A NIC supplies a computer's network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Data Communication and Computer Networking module | PDF page 14 | lines 32-49

Physical Layer:
• The Physical Layer provides a standardized interface to physical transmission media, including: Mechanical specification of electrical connectors and cables, for example maximum cable length, Electrical specification of transmission line and Bit-by-bit or symbol-by-symbol delivery.
• Interface: The Physical Layer defines the characteristics of interfaces between the devices & transmission medium.
• Representation of bits: The physical layer is concerned with transmission of signals from one device to another which involves converting data (1_s & 0_s) into signals and vice versa. It is not concerned with the meaning or interpretation of bits.
• Data rate: The physical layer defines the data

transmission rate i.e. number of bits sent per second. It is the responsibility of the physical layer to maintain the defined data rate.

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Data Communication and Computer Networking module | PDF page 16 | lines 18-32

Session Layer
• The session layer establishes a session between the communicating devices called dialog and synchronizes their interaction. It is the responsibility of the session layer to establish and synchronize the dialogs. It is also called the network dialog controller.
• The session layer at the sending side accepts data from the presentation layer adds checkpoints to it called sync bits and passes the data to the transport layer. At the receiving end the session layer receives data from the transport layer removes the checkpoints inserted previously and passes the data to the presentation layer. So, the main responsibility of session layer is dialog control and synchronization.

Question sources

Computer Science MCQ practice collection · Question 499

Question 196 | AM0500

Which protocol is used for translating domain names into IP addresses?

A. DNS

B. FTP

C. HTTP

D. SMTP

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A: DNS

DNS is the named external Domain Name Service, contrasted with host-address entries used without external naming. Evidence: PDF 18, 65.

Why each choice is right or wrong

A - Correct: DNS matches domain-to-address lookup (PDF 18, 65.)

B - Incorrect: FTP is file transfer (PDF 18, 65.)

C - Incorrect: HTTP is not the domain naming service (PDF 18, 65.)

D - Incorrect: SMTP is mail transfer (PDF 18, 65.)

Primary note proof

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

> Application Layer
> The Application Layer is a combination of Session,
> Presentation & Application Layers of OSI models and defines
> high level protocols like File Transfer (FTP), Electronic Mail
> (SMTP), Virtual Terminal (TELNET), Domain Name Service
> (DNS), etc.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

> /etc/hosts
> The /etc/hosts file always contains the localhost IP
> address, 127.0.0.1, which is
> usedfor interprocess communication.
> Never remove this line! Sometimes contains
> addresses of additional hosts, whichcan be contacted
> without using an external naming service such as
> DNS (the Domain Name Server).

Question sources

Computer Science MCQ practice collection · Question 500

Question 197 | AM0501

Which layer of the OSI reference model is responsible for establishing and terminating sessions between applications?

A. Presentation layer

B. Session layer

C. Transport layer

D. Application layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Session layer

The session layer establishes communication dialogs and controls/synchronizes their interaction, matching the explicitly stated sessions between applications. Evidence: PDF 16, 16, 16, 16.

Why each choice is right or wrong

A - Incorrect: Presentation performs representation translation (PDF 16, 16, 16, 16.)

B - Correct: Session establishes and synchronizes dialogs (PDF 16, 16, 16, 16.)

C - Incorrect: Transport connection setup is a different scope from the stated application sessions (PDF 16, 16, 16, 16.)

D - Incorrect: Application provides user access to network services (PDF 16, 16, 16, 16.)

Primary note proof

Study-note section NET19

OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET18

TCP establishes state with SYN, SYN+ACK, then ACK. SYN consumes one sequence position even with no application bytes. An acknowledgment states the next expected sequence number. For client ISN=395 and server ISN=260: client sends SYN seq395; server sends SYN+ACK seq260 ack396; client sends ACK seq396 ack261. Later byte counts advance sequence numbers. FIN also consumes one sequence position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 18-32

Session Layer
• The session layer establishes a session between the communicating devices called dialog and synchronizes their interaction. It is the responsibility of the session layer to establish and synchronize the dialogs. It is also called the network dialog controller.
• The session layer at the sending side accepts data from the presentation layer adds checkpoints to it called sync bits and passes the data to the transport layer. At the receiving end the session layer receives data from the transport layer removes the checkpoints inserted previously and passes the data to the presentation layer. So, the main responsibility of session layer is dialog control and synchronization.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.
• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.

• The main responsibility of the transport layer is process to process delivery of the entire message

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Question sources

Computer Science MCQ practice collection · Question 501

Question 198 | AM0502

A file-transfer client uses one connection for protocol commands and a separate connection for file data. Which listed application protocol uses this control/data separation?

A. SMTP

B. HTTP

C. FTP

D. TCP

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer C: FTP

FTP separates its control conversation from the data connection used for transfers. That architectural feature distinguishes it from the broad ability to download files, which HTTP also provides. SMTP handles mail submission and relay. TCP is the transport service on which an application protocol can build connections; it does not itself define FTP commands or the application's separate control and data roles.

Why each choice is right or wrong

A - Incorrect: SMTP defines mail transfer, not the described FTP control/data arrangement.

B - Incorrect: HTTP can transfer files, but does not define the described separate FTP control and data connections.

C - Correct: FTP uses separate control and data connections.

D - Incorrect: TCP provides transport connections; the application protocol defines their command/data roles.

Primary note proof

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

TCP - Transmission Control Protocol: TCP is a reliable
connection-oriented, reliable protocol. i.e. a connection is
established between the sender and receiver before the data can
be transmitted. It divides the data it receives from the upper
layer into segments and tags a sequence number to each
segment which is used at the receiving end for reordering of
data.
UDP - User Datagram Protocol: UDP is a simple protocol
used for process to process transmission. It is an unreliable,
connectionless protocol for applications that do not require flow
control or error control. It simply adds port address, checksum
and length information to the data it receives from the upper
layer.

Question sources

Computer Science MCQ practice collection · Question 502

Question 199 | AM0503

Which layer of the OSI reference model is responsible for providing services directly to the end-user applications?

A. Presentation layer

B. Session layer

C. Transport layer

D. Application layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Application layer

Application explicitly provides user access to network resources; the other layers perform representation, dialog and process-delivery work. Evidence: PDF 16, 16, 16, 16.

Why each choice is right or wrong

A - Incorrect: Presentation translates/encrypts/compresses (PDF 16, 16, 16, 16.)

B - Incorrect: Session establishes and synchronizes dialogs (PDF 16, 16, 16, 16.)

C - Incorrect: Transport segments and delivers between processes (PDF 16, 16, 16, 16.)

D - Correct: Application explicitly exposes network resources to users (PDF 16, 16, 16, 16.)

Primary note proof

Study-note section NET19

> OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 16 | lines 43-47

Application Layer
• The application layer enables the user to communicate its data to the receiver by providing certain services.
• The main Responsibility of Application layer is to provide the user access to network resources.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Data Communication and Computer Networking module | PDF page 16 | lines 18-32

Session Layer
• The session layer establishes a session between the communicating devices called dialog and synchronizes their interaction. It is the responsibility of the session layer to establish and synchronize the dialogs. It is also called the network dialog controller.
• The session layer at the sending side accepts data from the presentation layer adds checkpoints to it called sync bits and passes the data to the transport layer. At the receiving end the session layer receives data from the transport layer removes the checkpoints inserted previously and passes the data to the presentation layer. So, the main responsibility of session layer is dialog control and synchronization.

Data Communication and Computer Networking module | PDF page 16 | lines 2-17

• At the sending side, the transport layer receives data from the session layer, divides it into units called segments and sends it to the network layer.
• To ensure process to process delivery the transport layer makes use of port address to identify the data from the sending and receiving process.
• The data can be transported in a connection oriented or connectionless manner.
• The Transport layer is responsible for segmentation and reassembly of the message into segments which bear sequence numbers.

• The transport layer also carries out flow control and error control functions; but unlike data link layer these are end to end rather than node to node.
• The main responsibility of the transport layer is process to process delivery of the entire message

Question sources

Computer Science MCQ practice collection · Question 503

Question 200 | AM0504

Which subnet mask represents a Class C network?

A. 255.0.0.0

B. 255.255.0.0

C. 255.255.255.0

D. 255.255.255.255

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer C: 255.255.255.0, the historical Class C default

In the classful convention requested, Class C has a /24 default. Twenty-four leading one bits give three full 255 octets followed by zero: 255.255.255.0. This is a historical default rather than a restriction that every modern address in that range must use /24; CIDR can specify another prefix.

Why each choice is right or wrong

A - Incorrect: 255.0.0.0 is /8, the historical Class A default rather than Class C.

B - Incorrect: 255.255.0.0 is /16, the historical Class B default.

C - Correct: 255.255.255.0 is /24, matching the historical Class C default prefix.

D - Incorrect: 255.255.255.255 is /32, identifying one address rather than the Class C default network.

Primary note proof

Study-note section NET22

Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Study-note section NET24

A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide's 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Data Communication and Computer Networking module | PDF page 19 | lines 37-44

CIDR: A class B address is far too large for most organizationsand a class C network, with 256 addresses is too small. This leads to granting Class B address to organizations who do not require all. This is resulting in depletion of Address space. A solution is CIDR (Classless InterDomain Routing) The basic idea behind CIDR, is to allocate the remaining IP addresses in variable-sized blocks, without regard to the classes.

Question sources

Computer Science MCQ practice collection · Question 504

Question 201 | AM0505

What is the maximum number of hosts in a subnet with a subnet mask of 255.255.255.224?

A. 8

B. 16

C. 30

D. 62

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer C: 30 usable hosts

The mask 255.255.255.224 has 27 one bits and five host bits. Five bits yield 32 total addresses; reserving the subnet identifier and directed broadcast leaves 30 assignable ordinary hosts. This is an ordinary broadcast-subnet calculation, so the special /31 point-to-point exception is not relevant.

Why each choice is right or wrong

A - Incorrect: Eight does not equal the usable count from five host bits and two reserved addresses.

B - Incorrect: Sixteen is not the /27 usable-host count; this block has 32 total addresses.

C - Correct: A /27 has 32 total addresses and 30 ordinary assignable host addresses.

D - Incorrect: Sixty-two usable hosts requires six host bits, corresponding to /26 rather than /27.

Primary note proof

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Study-note section NET24

> A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide's 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

> Address Space: An address space is the total
> number ofaddresses used by the protocol.
> • If a protocol uses N bits to define an address,
> the address space is 2N
> because each bit can
> have twodifferent values (0 or 1) and N bits
> can have 2N values.
> • IPv4 uses 32-bit addresses, which means that
> the address space is 232 or 4,294,967,296
> (more than 4billion).
> Notations: There are two notations to show an IPv4 address:
> • Binary notation: The IPv4 address is displayed

as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science MCQ practice collection · Question 505

Question 202 | AM0506

Which IP address belongs to the loopback network?

A. 127.0.0.1

B. 192.168.0.1

C. 10.0.0.1

D. 172.16.0.1

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer A: 127.0.0.1

The module explicitly identifies 127.0.0.1 as localhost for interprocess communication. This supports the loopback instance, without needing an unprovided full 127/8 range rule. Evidence: PDF 65, 19.

Why each choice is right or wrong

A - Correct: 127.0.0.1 is explicitly localhost (PDF 65, 19.)

B - Incorrect: 192.168.0.1 is not the cited localhost address (PDF 65, 19.)

C - Incorrect: 10.0.0.1 is not the cited localhost address (PDF 65, 19.)

D - Incorrect: 172.16.0.1 is not the cited localhost address (PDF 65, 19.)

Primary note proof

Study-note section NET23

> RFC1918 private blocks are 10.0.0.0/8, 172.16.0.0/12 (through 172.31.255.255), and 192.168.0.0/16. They can be reused internally but are not globally routed as public destinations. A LAN may also use legitimately assigned public addresses; private is common, not mandatory. 127.0.0.0/8 is IPv4 loopback, commonly 127.0.0.1, for local protocol access without an external link. 255.255.255.255 is limited broadcast; 0.0.0.0 can identify an unconfigured source. Link-local/APIPA uses 169.254.0.0/16, with edge /24 blocks excluded from normal automatic selection.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

> /etc/hosts
> The /etc/hosts file always contains the localhost IP
> address, 127.0.0.1, which is
> usedfor interprocess communication.
> Never remove this line! Sometimes contains
> addresses of additional hosts, whichcan be contacted
> without using an external naming service such as
> DNS (the Domain Name Server).

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

> NAT (Network Address Translation): The scarcity of
> network addresses in IPv4 led to the development of IPv6.
> Due to this, we need to use private IP address inside the
> organization and translate it to public IP address using
> NAT.

Question sources

Computer Science MCQ practice collection · Question 506

Question 203 | AM0507

How many subnets can be created from a Class B network with a subnet mask of 255.255.240.0?

A. 8

B. 16

C. 32

D. 64

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: 16 subnets

The question explicitly supplies a Class B parent, so use its historical /16. The given mask 255.255.240.0 is /20. Borrowing 20−16=4 bits yields 2^4=16 equal subnets under modern practice. The note says not to subtract two subnets; host capacity is a different calculation.

Why each choice is right or wrong

A - Incorrect: Eight subnets would borrow three bits and produce /19 from a /16 parent.

B - Correct: The /20 mask borrows four bits from /16, yielding sixteen equal subnets.

C - Incorrect: Thirty-two subnets would borrow five bits and require /21 from the Class B parent.

D - Incorrect: Sixty-four subnets would borrow six bits and require /22 rather than /20.

Primary note proof

Study-note section NET26

> To split parent /a into equal /p blocks, borrow b=p-a bits and obtain 2^b subnets. Do not subtract two subnets under modern practice. For a Class B /16 parent split to /19, b=3 gives eight subnets, each with 8192 addresses and 8190 conventional hosts. A /19 alone does not state its parent.

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Study-note section NET24

> A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP
addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Question sources

Computer Science MCQ practice collection · Question 507

Question 204 | AM0508

What is the default subnet mask for a Class A network?

A. 255.0.0.0

B. 255.255.0.0

C. 255.255.255.0

D. 255.255.255.255

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer A: 255.0.0.0

The historical Class A default is /8. Eight leading mask ones fill only the first octet, giving 255.0.0.0. The other choices correspond to /16, /24 and /32; they can be explicit CIDR prefixes, but they are not the historical default asked for.

Why each choice is right or wrong

A - Correct: 255.0.0.0 contains eight leading ones and is the historical Class A default.

B - Incorrect: 255.255.0.0 is /16, the historical default for Class B.

C - Incorrect: 255.255.255.0 is /24, the historical default for Class C.

D - Incorrect: 255.255.255.255 is /32 and describes one address, not the Class A default.

Primary note proof

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Study-note section NET24

> A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

> Classful addressing : In classful addressing, the
> addressspace is divided into five classes: A, B, C,
> D, and E.
> • Netid and Hostid In classful addressing, an IP

addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Question sources

Computer Science MCQ practice collection · Question 508

Question 205 | AM0509

For a conventional IPv4 link needing two usable endpoints and reserving network/broadcast addresses, which offered mask allocates the smallest sufficient block?

A. 255.255.255.0

B. 255.255.255.128

C. 255.255.255.192

D. 255.255.255.252

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer D: 255.255.255.252

The offered masks correspond to /24, /25, /26 and /30 blocks. A conventional /30 contains four addresses, leaving two usable endpoints after reserving network and broadcast addresses. The other choices allocate more addresses than this link needs. Supported /31 point-to-point links are a separate exception, so the question limits the comparison to the conventional model and the masks actually offered.

Why each choice is right or wrong

A - Incorrect: /24 is sufficient but allocates 256 addresses, more than the smallest offered sufficient block.

B - Incorrect: /25 allocates 128 addresses and is not the smallest sufficient offered block.

C - Incorrect: /26 allocates 64 addresses and is not the smallest sufficient offered block.

D - Correct: /30 allocates four addresses and two conventional usable endpoints, the smallest sufficient offered block.

Primary note proof

Study-note section NET28

> A conventional /30 has four addresses and two usable endpoints; check the link and addressing assumptions when selecting a WAN prefix. RFC3021 permits /31 on supported point-to-point links: both addresses are endpoints, so the subtract-two rule does not apply. /32 identifies one address/host route. Do not claim /30 is universally the smallest valid link prefix.

Study-note section NET24

> A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP
addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Data Communication and Computer Networking module | PDF page 11 | lines 34-51

NETWORK TOPOLOGY
A network topology is a description of the layout of the region
or area covered by that network.
There are two types of connections. They are: point-to-point
and multi-point.
Point-to-point connections provide a direct link between two
devices.
Multi-point connections provide a link between three or more

devices on a network.
There are seven basic topologies in the study of network topology:
• Point-to-point topology,
• Bus (point-to-multipoint) topology,
• Ring topology,
• Star topology,
• Hybrid topology,
• Mesh topology and,
• Tree topology.

Question sources

Computer Science MCQ practice collection · Question 509

Question 206 | AM0510

How many bits are in the host portion of an IPv4 address with a subnet mask of 255.255.255.192?

A. 6

B. 8

C. 10

D. 14

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer A: 6

192=11000000 leaves six zero mask bits; the full mask is /26. Hence 32-26=6 host bits. Evidence: PDF 18, 19, 19.

Why each choice is right or wrong

A - Correct: Six is the calculated count (PDF 18, 19, 19.)

B - Incorrect: Eight would correspond to /24 (PDF 18, 19, 19.)

C - Incorrect: Ten would correspond to /22 (PDF 18, 19, 19.)

D - Incorrect: Fourteen would correspond to /18 (PDF 18, 19, 19.)

Primary note proof

Study-note section NET24

A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide's 255.248.0.0 mistake.

Study-note section NET25

For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science MCQ practice collection · Question 510

Question 207 | AM0511

What is the network address for the IP address 192.168.10.25/28?

A. 192.168.10.0

B. 192.168.10.16

C. 192.168.10.24

D. 192.168.10.25

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: 192.168.10.16

/28 mask last octet is 240. 25=00011001 AND 240=11110000 gives 16=00010000. Therefore network address is 192.168.10.16. Evidence: PDF 18, 19, 19.

Why each choice is right or wrong

A - Incorrect: 0 is the preceding /28 block (PDF 18, 19, 19.)

B - Correct: 16 is the correct masked boundary (PDF 18, 19, 19.)

C - Incorrect: 24 retains host bits and is not a 16-step boundary (PDF 18, 19, 19.)

D - Incorrect: 25 is the given host address, not the masked network (PDF 18, 19, 19.)

Primary note proof

Study-note section NET24

Compute network=address AND mask. Host bits zero identify the block start; host bits one give broadcast. With 172.16.28.252/20, the third octet is grouped in blocks of 16, so the network is 172.16.16.0 and broadcast 172.16.31.255. A written host/prefix such as 172.105.180.0/21 need not be an aligned block label: masking yields 172.105.176.0/21.

Study-note section NET25

192.168.2.64/26 spans .64-.127; usable .65-.126 gives 62 hosts. 192.168.10.25/28 spans .16-.31, so its network is .16. 172.17.128.0/21 spans 172.17.128.0-172.17.135.255; .128.255 and .135.0 are valid interior hosts, but .135.255 is broadcast. /27 gives 32 total and 30 usable, not 32 usable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)

is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science MCQ practice collection · Question 511

Question 208 | AM0512

How many usable host addresses are available in a subnet with a subnet mask of 255.255.255.224?

A. 6

B. 14

C. 30

D. 62

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer C: 30 usable host addresses

224 in the final mask octet is 11100000. That yields /27 and h=5. Computing 2^5−2 gives 30 ordinary usable host addresses after network and directed-broadcast reservations. The distractors correspond to different host-bit counts; none changes the explicit mask in this question.

Why each choice is right or wrong

A - Incorrect: Six usable hosts would use three host bits, normally /29 rather than /27.

B - Incorrect: Fourteen usable hosts would use four host bits, normally /28.

C - Correct: Five host bits give 32 addresses minus two reservations, yielding 30 usable hosts.

D - Incorrect: Sixty-two usable hosts would use six host bits and a /26 mask.

Primary note proof

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Study-note section NET24

> A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide's 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

> Address Space: An address space is the total
> number ofaddresses used by the protocol.
> • If a protocol uses N bits to define an address,
> the address space is 2N
> because each bit can
> have twodifferent values (0 or 1) and N bits
> can have 2N values.
> • IPv4 uses 32-bit addresses, which means that
> the address space is 232 or 4,294,967,296
> (more than 4billion).
> Notations: There are two notations to show an IPv4 address:
> • Binary notation: The IPv4 address is displayed
> as 32bits. ex. 11000001 10000011 00011011
> 11111111
> • Dotted decimal notation: To make the IPv4
> address easier to read, Internet addresses are
> usually written in decimal form with a decimal
> point (dot) separatingthe bytes. Each byte (octet)
> is 8 bits hence each number in dotted-decimal
> notation is a value rangingfrom 0 to 255. Ex.
> 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science MCQ practice collection · Question 512

Question 209 | AM0513

Which address category is defined for internal reuse by RFC1918 rather than global public routing?

A. Public IP address

B. Private IP address

C. Reserved IP address

D. Dynamic IP address

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B: Private IP address

RFC1918 defines private blocks that organizations can reuse internally and that are not globally routed as public destinations. Public addresses can also be used on a LAN when legitimately assigned, so local communication alone would not distinguish the categories. Dynamic describes an assignment method such as a DHCP lease, not an address scope. The revised scope identifies the category by its defined allocation purpose.

Why each choice is right or wrong

A - Incorrect: Public addresses are globally allocated unicast space rather than RFC1918 internal-reuse space.

B - Correct: Private addresses are the RFC1918 blocks defined for internal reuse.

C - Incorrect: Reserved addresses have specific purposes and are not a generic ordinary-LAN host address category.

D - Incorrect: Dynamic describes how an address is assigned, not whether it belongs to RFC1918 space.

Primary note proof

Study-note section NET23

RFC1918 private blocks are 10.0.0.0/8, 172.16.0.0/12 (through 172.31.255.255), and 192.168.0.0/16. They can be reused internally but are not globally routed as public destinations. A LAN may also use legitimately assigned public addresses; private is common, not mandatory. 127.0.0.0/8 is IPv4 loopback, commonly 127.0.0.1, for local protocol access without an external link. 255.255.255.255 is limited broadcast; 0.0.0.0 can identify an unconfigured source. Link-local/APIPA uses 169.254.0.0/16, with edge /24 blocks excluded from normal automatic selection.

Study-note section NET20

DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of
network addresses in IPv4 led to the development of IPv6.
Due to this, we need to use private IP address inside the
organization and translate it to public IP address using
NAT.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local
network to a single IP address. This address is often used
by the router that connects the computers to the Internet.
The router can be connected to a DSL modem, cable
modem, T1 line, or even a dial-up modem. When other
computers on the Internet attempt to access computers
within the local network, they only see the IP address of
the router. This adds an extra level of security, since the
router can be configured as a firewall, only allowing
authorized systems to access the computers within the
network.

Question sources

Computer Science MCQ practice collection · Question 513

Question 210 | AM0514

Which of the following is not a valid IP address?

A. 192.168.0.1

B. 10.0.0.256

C. 172.16.0.0

D. 255.255.255.0

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer B: 10.0.0.256

Each IPv4 decimal octet must be 0-255. 10.0.0.256 violates that constraint; the other strings are syntactically valid IPv4 values regardless of host-assignability restrictions. Evidence: PDF 19.

Why each choice is right or wrong

A - Incorrect: All four octets satisfy the syntax rule (PDF 19.)

B - Correct: 256 exceeds the eight-bit maximum (PDF 19.)

C - Incorrect: All octets are syntactically valid; a network-address role is a separate issue (PDF 19.)

D - Incorrect: All octets are syntactically valid; use as a mask does not make the bit string invalid (PDF 19.)

Primary note proof

Study-note section NET22

> Write IPv4 as four decimal octets 0-255, or as 32 binary bits. Use weights 128,64,32,16,8,4,2,1 within each octet. Thus 11000001 10000011 00011011 11111111 is 193.131.27.255. A value such as 257 is impossible. Syntactic validity differs from assignability: an address ending .0 or .255 can be a host under some prefixes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

> Address Space: An address space is the total
> number ofaddresses used by the protocol.
> • If a protocol uses N bits to define an address,
> the address space is 2N
> because each bit can
> have twodifferent values (0 or 1) and N bits
> can have 2N values.
> • IPv4 uses 32-bit addresses, which means that

the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Question sources

Computer Science MCQ practice collection · Question 514

Question 211 | AM0515

What is the default subnet mask for a Class B IP address?

A. 255.0.0.0

B. 255.255.0.0

C. 255.255.255.0

D. 255.255.255.255

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: 255.255.0.0

NET22 gives historical Class B a /16 default. The first sixteen mask bits are ones, producing 255.255.0.0. This answers the requested default-mask question; it does not imply that a modern Class B-range address cannot have a different explicit CIDR prefix.

Why each choice is right or wrong

A - Incorrect: 255.0.0.0 is /8, the historical default for Class A.

B - Correct: 255.255.0.0 has sixteen leading ones and is the historical Class B default.

C - Incorrect: 255.255.255.0 is /24, the historical default for Class C.

D - Incorrect: 255.255.255.255 is /32, a single-address prefix rather than a Class B default.

Primary note proof

Study-note section NET22

Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Study-note section NET24

A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address, the address space is 2N because each bit can have twodifferent values (0 or 1) and N bits can have 2N values.
• IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal

point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Question sources

Computer Science MCQ practice collection · Question 515

Question 212 | AM0516

How many bits are used for the host portion of an IP address with a subnet mask of 255.255.255.0?

A. 8 bits

B. 16 bits

C. 24 bits

D. 32 bits

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer A: 8 bits

255.255.255.0 contains 24 one bits and eight zero host bits. Subtract the 24-bit prefix from the 32-bit address width. Evidence: PDF 18, 19, 19.

Why each choice is right or wrong

A - Correct: Eight is 32-24 (PDF 18, 19, 19.)

B - Incorrect: Sixteen would require a /16 prefix (PDF 18, 19, 19.)

C - Incorrect: Twenty-four is the prefix count, not the host count (PDF 18, 19, 19.)

D - Incorrect: Thirty-two is the entire address width (PDF 18, 19, 19.)

Primary note proof

Study-note section NET24

A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Study-note section NET25

For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet’s increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the Internet. Every host and router on the Internet has an IP address, which encodes its network number and host number.
• The combination is unique: in principle, no two machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination address fields of IP packets.
• An IP address does not refer to a host but it refers to a network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address, the address space is 2N because each bit can have twodifferent values (0 or 1) and N bits can have 2N values.
• IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to

the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science MCQ practice collection · Question 516

Question 213 | AM0517

Given a subnet mask of 255.255.255.224, how many host addresses are available in each subnet?

A. 6

B. 14

C. 30

D. 62

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer C: 30 host addresses per ordinary subnet

The mask 255.255.255.224 is /27, leaving five host bits. A block therefore contains 32 addresses. Because the question asks addresses available to hosts, exclude its network and broadcast addresses, leaving 30. Treating all 32 as usable would confuse block size with conventional broadcast-LAN host capacity.

Why each choice is right or wrong

A - Incorrect: Six usable addresses corresponds to /29, not the stated /27 mask.

B - Incorrect: Fourteen usable addresses corresponds to /28 and four host bits.

C - Correct: The /27 block has five host bits and 32−2=30 conventional usable addresses.

D - Incorrect: Sixty-two usable addresses would require /26 and six host bits.

Primary note proof

Study-note section NET25

For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet’s increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Study-note section NET24

A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP
addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Question sources

Computer Science MCQ practice collection · Question 517

Question 214 | AM0518

What is the network address for the IP address 192.168.1.65/26?

A. 192.168.1.0

B. 192.168.1.64

C. 192.168.1.65

D. 192.168.1.128

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: 192.168.1.64

/26 makes the fourth-octet mask192=11000000. 65=01000001 AND192=11000000 gives64=01000000, so the network is 192.168.1.64. Evidence: PDF 18, 19, 19.

Why each choice is right or wrong

A - Incorrect: 0 is the preceding 64-address block (PDF 18, 19, 19.)

B - Correct: 64 is the masked network boundary (PDF 18, 19, 19.)

C - Incorrect: 65 retains the nonzero host bit (PDF 18, 19, 19.)

D - Incorrect: 128 is the next block boundary (PDF 18, 19, 19.)

Primary note proof

Study-note section NET24

> Compute network=address AND mask. Host bits zero identify the block start; host bits one give broadcast. With 172.16.28.252/20, the third octet is grouped in blocks of 16, so the network is 172.16.16.0 and broadcast 172.16.31.255. A written host/prefix such as 172.105.180.0/21 need not be an aligned block label: masking yields 172.105.176.0/21.

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the Internet. Every host and router on the Internet has an IP address, which encodes its network number and host number.
• The combination is unique: in principle, no two machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination address fields of IP packets.
• An IP address does not refer to a host but it refers to a network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address, the address space is 2N because each bit can have twodifferent values (0 or 1) and N bits can have 2N values.
• IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed as 32bits. ex. 11000001 10000011 00011011 11111111
• Dotted decimal notation: To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separatingthe bytes. Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value rangingfrom 0 to 255. Ex. 129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the addressspace is divided into five classes: A, B, C, D, and E.
• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex.

255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science MCQ practice collection · Question 518

Question 215 | AM0519

How many subnets can be created from a Class C network with a subnet mask of 255.255.255.192?

A. 2

B. 4

C. 8

D. 16

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: 4 subnets

A historical Class C parent is /24. The supplied 255.255.255.192 mask is /26, borrowing two bits. Modern counting gives 2^2=4 equal subnets, not two: NET26 explicitly rejects subtracting two subnets under modern practice. Here the parent is given, unlike the under-specified standalone-prefix questions.

Why each choice is right or wrong

A - Incorrect: Two would incorrectly exclude two subnets under an obsolete unstated convention; modern counting gives four.

B - Correct: Borrowing two bits from /24 to /26 produces four equal subnets.

C - Incorrect: Eight subnets requires borrowing three bits, producing /27 rather than /26.

D - Incorrect: Sixteen subnets requires four borrowed bits and a /28 prefix.

Primary note proof

Study-note section NET26

To split parent /a into equal /p blocks, borrow b=p-a bits and obtain 2^b subnets. Do not subtract two subnets under modern practice. For a Class B /16 parent split to /19, b=3 gives eight subnets, each with 8192 addresses and 8190 conventional hosts. A /19 alone does not state its parent.

Study-note section NET22

Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have

special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.
• Netid and Hostid In classful addressing, an IP
addressin class A, B, or C is divided into netid
and hostid.
Subnetting: It allows a network to be split into several
parts for internal use but still act like a single network to
the outsideworld.
• To implement subnetting, the router needs a
subnet mask that indicates the split between
network + subnet number and host. Ex.
255.255.252.0/22. A‖/22‖ to indicate that the
subnet mask is 22 bits long.

Data Communication and Computer Networking module | PDF page 19 | lines 37-44

CIDR: A class B address is far too large for most organizationsand a class C network, with 256 addresses is too small. This leads to granting Class B address to organizations who do not require all. This is resulting in depletion of Address space. A solution is CIDR (Classless InterDomain Routing) The basic idea behind CIDR, is to allocate the remaining IP addresses in variable-sized blocks, without regard to the classes.

Question sources

Computer Science MCQ practice collection · Question 519

Question 216 | AM0520

What is the maximum number of IP addresses that can be assigned to hosts in a subnet with a /27 subnet mask?

A. 14

B. 30

C. 62

D. 126

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: 30 assignable host IP addresses

With /27, 32−27=5 host bits produce 2^5=32 addresses. For an ordinary subnet, network and broadcast are reserved, so 30 can be assigned to hosts. The alternative values are usable-host counts for /28, /26 and /25, rather than the supplied /27.

Why each choice is right or wrong

A - Incorrect: Fourteen usable hosts requires /28, leaving only four host bits.

B - Correct: A /27 leaves five host bits and therefore 30 assignable ordinary host addresses.

C - Incorrect: Sixty-two usable hosts is the /26 result, not /27.

D - Incorrect: 126 usable hosts requires /25 and seven host bits.

Primary note proof

Study-note section NET25

For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet’s increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Study-note section NET24

A /p IPv4 prefix has p leading one bits followed by 32-p zeros. Partial octet masks are 128,192,224,240,248,252,254. /19 is 255.255.224.0; /20 is 255.255.240.0; /21 is 255.255.248.0, correcting the study guide’s 255.248.0.0 mistake.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal
notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Data Communication and Computer Networking module | PDF page 19 | lines 23-36

Classful addressing : In classful addressing, the
addressspace is divided into five classes: A, B, C,
D, and E.

• Netid and Hostid In classful addressing, an IP addressin class A, B, or C is divided into netid and hostid.
Subnetting: It allows a network to be split into several parts for internal use but still act like a single network to the outsideworld.
• To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A‖/22‖ to indicate that the subnet mask is 22 bits long.

Question sources

Computer Science MCQ practice collection · Question 520

Question 217 | AM0521

Which IP address range is reserved for Automatic Private IP Addressing (APIPA)?

A. 10.0.0.0 - 10.255.255.255

B. 127.0.0.0 - 127.255.255.255

C. 169.254.0.0 - 169.254.255.255

D. 192.168.0.0 - 192.168.255.255

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer C: 169.254.0.0/16 link-local/APIPA range

NET23 assigns link-local/APIPA to 169.254.0.0/16, the range printed in C. That identifies the reserved overall block; the first and last /24 blocks are excluded from normal automatic selection. The private RFC1918 blocks and loopback block offered elsewhere have different purposes and are not the APIPA range.

Why each choice is right or wrong

A - Incorrect: 10/8 is RFC1918 private space, not the link-local/APIPA allocation.

B - Incorrect: 127/8 is loopback for local protocol access, not automatic link-local addressing.

C - Correct: 169.254/16 is the APIPA block; normal automatic host selection excludes its edge /24 blocks.

D - Incorrect: 192.168/16 is RFC1918 private space rather than the APIPA block.

Primary note proof

Study-note section NET23

RFC1918 private blocks are 10.0.0.0/8, 172.16.0.0/12 (through 172.31.255.255), and 192.168.0.0/16. They can be reused internally but are not globally routed as public destinations. A LAN may also use legitimately assigned public addresses; private is common, not mandatory. 127.0.0.0/8 is IPv4 loopback, commonly 127.0.0.1, for local protocol access without an external link. 255.255.255.255 is limited broadcast; 0.0.0.0 can identify an unconfigured source. Link-local/APIPA uses 169.254.0.0/16, with edge /24 blocks excluded from normal automatic selection.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of
network addresses in IPv4 led to the development of IPv6.
Due to this, we need to use private IP address inside the
organization and translate it to public IP address using
NAT.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

/etc/hosts
The /etc/hosts file always contains the localhost IP
address, 127.0.0.1, which is
usedfor interprocess communication.
Never remove this line! Sometimes contains
addresses of additional hosts, whichcan be contacted
without using an external naming service such as
DNS (the Domain Name Server).

Data Communication and Computer Networking module | PDF page 19 | lines 2-22

Address Space: An address space is the total
number ofaddresses used by the protocol.
• If a protocol uses N bits to define an address,
the address space is 2N
because each bit can
have twodifferent values (0 or 1) and N bits
can have 2N values.
• IPv4 uses 32-bit addresses, which means that
the address space is 232 or 4,294,967,296
(more than 4billion).
Notations: There are two notations to show an IPv4 address:
• Binary notation: The IPv4 address is displayed
as 32bits. ex. 11000001 10000011 00011011
11111111
• Dotted decimal notation: To make the IPv4
address easier to read, Internet addresses are
usually written in decimal form with a decimal
point (dot) separatingthe bytes. Each byte (octet)
is 8 bits hence each number in dotted-decimal

notation is a value rangingfrom 0 to 255. Ex.
129.11.11.239

Question sources

Computer Science MCQ practice collection · Question 521

Question 218 | AM0523

Which of the following protocols is responsible for assigning IP addresses to hosts dynamically?

A. DNS

B. DHCP

C. HTTP

D. FTP

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer B: DHCP

DHCP is the address-configuration service in NET20, dynamically leasing an address together with other settings. DNS resolves names rather than assigning a client's host address. HTTP and FTP handle web exchanges and file transfers. Therefore B matches dynamic assignment rather than name resolution or application-data transfer.

Why each choice is right or wrong

A - Incorrect: DNS resolves names to records and does not perform the dynamic lease assignment asked about.

B - Correct: DHCP dynamically leases host addresses and related network configuration.

C - Incorrect: HTTP provides web request/response exchanges rather than dynamic IP-address allocation.

D - Incorrect: FTP transfers files and is not a host-address lease protocol.

Primary note proof

Study-note section NET20

DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET21

HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 32-45

Application Layer
The application layer in TCP/IP is equivalent to the combined session, presentation, and application layers in the OSI model and many protocols are defined at this layer.
The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session, Presentation & Application Layers of OSI models and defines high level protocols like File Transfer (FTP), Electronic Mail (SMTP), Virtual Terminal (TELNET), Domain Name Service (DNS), etc.

Question sources

Computer Science MCQ practice collection · Question 523

Question 219 | CS0155

Which TCP/IP-suite protocols listed below are connectionless? Select all that apply.

A. User Datagram Protocol (UDP)

B. Internet Protocol

C. Transmission Control Protocol

D. The OSI reference protocol

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A, B: User Datagram Protocol (UDP); Internet Protocol

UDP sends independent datagrams without built-in connection setup, ordering or retransmission. IP likewise provides connectionless best-effort packet delivery at the network layer. TCP instead establishes transport connection state and supplies a reliable ordered stream. OSI is a reference model rather than a protocol in the TCP/IP suite. The multiple-selection instruction therefore permits both connectionless protocols across the two layers.

Why each choice is right or wrong

A - Correct: UDP has no built-in connection setup and treats datagrams independently.

B - Correct: IP also supplies connectionless best-effort packet delivery at the network layer.

C - Incorrect: TCP establishes logical transport connection state before its reliable stream transfer.

D - Incorrect: OSI is a reference model, not a connectionless TCP/IP-suite protocol.

Primary note proof

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET16

> IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Study-note section NET03

> A reference model divides responsibilities and interfaces; it is not itself executable networking software. OSI is a seven-layer reference model; TCP/IP describes an implemented protocol suite, usually taught with four layers. A five-layer teaching model separates physical and link functions. State the model before counting layers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 9-21

> TCP - Transmission Control Protocol: TCP is a reliable
> connection-oriented, reliable protocol. i.e. a connection is
> established between the sender and receiver before the data can

be transmitted. It divides the data it receives from the upper layer into segments and tags a sequence number to each segment which is used at the receiving end for reordering of data.
UDP - User Datagram Protocol: UDP is a simple protocol used for process to process transmission. It is an unreliable, connectionless protocol for applications that do not require flow control or error control. It simply adds port address, checksum and length information to the data it receives from the upper layer.

Data Communication and Computer Networking module | PDF page 17 | lines 22-38

Network Layer or IP
Also called as the Internetwork Layer (IP). It holds the IP protocol which is a network layer protocol and is responsiblefor source to destination transmission of data.
The Internetworking Protocol (IP) is a connection-less & unreliable protocol.
It is a best effort delivery service. i.e. there is no error checking in IP, it simply sends the data and relies on its underlying layers to get the data transmitted to the destination.Even if it is unreliable IP cannot be considered weak and useless; since it provides only the functionality that is requiredfor transmitting data thereby giving maximum efficiency.
Since there is no mechanism of error detection or correction in IP, there will be no delay introduced on a medium where thereis no error at all.

Data Communication and Computer Networking module | PDF page 14 | lines 12-29

The Open Systems Interconnection (OSI) Model was developed by International Organization for Standardization (ISO). ISO is the organization, OSI is the model
It was developed to allow systems with different platforms (Hardware & software) to communicate with each other.
Now, it is a network model that defines the protocols for network communications.
It is a hierarchical model that groups its processes into seven (7) layers. (Top to Bottom)
1. Application Layer
2. Presentation Layer
3. Session Layer
4. Transport Layer
5. Network Layer
6. Data Link Layer

7. Physical Layer
Each layer has specific functions to perform and has to co-operate with the layers above and below it.

Question sources

Computer Science Exit Exam practice collection · Question 155

Question 220 | CS0156

Which statements describe capabilities or properties of IPv6? Select all that apply. Encryption support in D means capability, not automatic confidentiality or exclusivity to IPv6.

A. Handle more addresses

B. Use 128-bits

C. Route slower and is memory intensive

D. Support encryptions

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer A, B, D: Handle more addresses; Use 128-bits; Support encryptions

IPv6 uses 128-bit addresses, giving it a much larger address space than IPv4's 32-bit addresses. It can support encryption, but that capability does not mean every IPv6 packet is encrypted or that IPv4 cannot support encryption. Routing speed and memory requirements depend on implementation and workload; they are not inherently fixed by the protocol version. Thus the three capability/property statements qualify.

Why each choice is right or wrong

A - Correct: A 128-bit address space can represent far more addresses than IPv4's 32-bit space.

B - Correct: NET23 explicitly states that IPv6 addresses contain 128 bits.

C - Incorrect: Slower routing and greater memory use are not inherent requirements of every IPv6 implementation.

D - Correct: Encryption is supported, but IPv6 does not automatically encrypt traffic and this is not unique to IPv6.

Primary note proof

Study-note section NET23

IPv6 has 128-bit hexadecimal addresses, flexible prefixes and no IPv4-style broadcast. Sources, not routers, fragment IPv6 packets. IPv6 does not automatically encrypt traffic or make authentication unnecessary. Compare IPv6's larger 128-bit address space and encryption support separately; specify the property being evaluated.

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 45-54

IP addresses: IP address uniquely identifies a device on the
Internet. Every host and router on the Internet has an IP address,
which encodes its network number and host number.
• The combination is unique: in principle, no two
machines on the Internet have the same IP address.
• An IPv4 address is 32 bits long
• They are used in the Source address and Destination
address fields of IP packets.
• An IP address does not refer to a host but it refers to a
network interface.

Data Communication and Computer Networking module | PDF page 19 | lines 45-49

NAT (Network Address Translation): The scarcity of
network addresses in IPv4 led to the development of IPv6.
Due to this, we need to use private IP address inside the
organization and translate it to public IP address using
NAT.

Question sources

Computer Science Exit Exam practice collection · Question 156

Question 221 | CS0157

Which one does NOT describe a loopback address?

A. It helps to refer to the host itself

B. It allows programs to access network protocols for local services

C. It is an external pseudo address

D. It is denoted by the IP address: 127.0.0.1

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer C: Loopback is local self-reference, not an external address

Loopback lets a host address itself and access local services through networking protocols without an external link. NET23 identifies 127/8 and common 127.0.0.1 for that purpose. Describing it as external contradicts this local scope, whereas the remaining choices all describe supported loopback uses or notation.

Why each choice is right or wrong

A - Incorrect: Referring to the host itself is precisely the loopback function, so this statement does describe it.

B - Incorrect: Local programs can use networking protocols to access local services through loopback.

C - Correct: External contradicts loopback's local self-reference without an external network link.

D - Incorrect: 127.0.0.1 is explicitly given as the common IPv4 loopback address.

Primary note proof

Study-note section NET23

> RFC1918 private blocks are 10.0.0.0/8, 172.16.0.0/12 (through 172.31.255.255), and 192.168.0.0/16. They can be reused internally but are not globally routed as public destinations. A LAN may also use legitimately assigned public addresses; private is common, not mandatory. 127.0.0.0/8 is IPv4 loopback, commonly 127.0.0.1, for local protocol access without an external link. 255.255.255.255 is limited broadcast; 0.0.0.0 can identify an unconfigured source. Link-local/APIPA uses 169.254.0.0/16, with edge /24 blocks excluded from normal automatic selection.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

> /etc/hosts
> The /etc/hosts file always contains the localhost IP
> address, 127.0.0.1, which is
> usedfor interprocess communication.
> Never remove this line! Sometimes contains
> addresses of additional hosts, whichcan be contacted
> without using an external naming service such as
> DNS (the Domain Name Server).

Question sources

Computer Science Exit Exam practice collection · Question 157

Question 222 | CS0164

In data communication, which of the following is NOT a factor for data rate?

A. Travel distance

B. Bandwidth available

C. Level of noise

D. Levels of the signals

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer A: Travel distance is not explicit in the simplified capacity formulas

In the simplified capacity comparison taught in NET05, Nyquist uses bandwidth and signal levels, while Shannon uses bandwidth and signal-to-noise ratio. Travel distance is not a separate formula parameter, making A the intended choice. The qualification matters: distance can affect attenuation, delay and real achievable rates, so the answer is not physical independence from distance.

Why each choice is right or wrong

A - Correct: Travel distance is absent as an independent variable in the taught formulas, despite indirect real-link effects.

B - Incorrect: Bandwidth appears directly in both Nyquist and Shannon expressions and limits channel capacity.

C - Incorrect: Noise enters Shannon capacity through the signal-to-noise ratio.

D - Incorrect: Signal levels enter Nyquist capacity through the number of bits represented per symbol.

Primary note proof

Study-note section NET05

> For an ideal noiseless channel, Nyquist gives $R \leq 2B \log_2(M)$. For Gaussian noise, Shannon gives $C = B \log_2(1+S/N)$, with a linear power ratio; $SNR(dB) = 10 \log_{10}(S/N)$. Bandwidth, signal levels and noise appear explicitly. Distance affects attenuation, delay and often achievable rate, but is not a separate variable in these simplified capacity formulas.

Study-note section NET06

> Transmission delay is packet bits divided by link bit rate; propagation delay is distance divided by signal speed. Processing and queueing add delay. A 1000-byte packet takes 0.8 ms to serialize at 10 Mbit/s; increasing line rate does not reduce propagation time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 9 | lines 2-14

BIT LENGTH or Bit Interval (Tb)
It is the time required to send one bit.
It is measured in seconds.
BIT RATE
It is the number of bits transmitted in one second.
It is expressed as bits per second (bps).
Relation between bit rate and bit interval can be as follows
Bit rate = 1 / Bit interval
Baud Rate
It is the rate of Signal Speed, i.e the rate at which the signal changes.
A digital signal with two levels 0_& 1_will have the same baud rate and bit rate & bit rate.

Data Communication and Computer Networking module | PDF page 10 | lines 18-35

receiver.There are three causes of impairment:
• Attenuation
• Distortion
• Noise
Attenuation
Attenuation results in loss of energy due to distance.
Whena signal travels through a medium, it loses some of its energy in overcoming the resistance of the medium. The electrical energy in the signal may convert to heat. To compensate for this loss, amplifiers are used to amplify thesignal.
Distortion
Distortion changes the shape of the signal.
Noise
Noise is any unwanted signal that is mixed or combined with the original signal during transmission.
Due to noise the original signal is altered and signal received is not sameas the one sent.

Data Communication and Computer Networking module | PDF page 9 | lines 29-49

Transmission of Digital signal: Digital signal can be transmitted in the following two ways:
• Baseband Transmission: The signal is transmitted without making any change to it (i.e. Without modulation)
o In baseband transmission, the bandwidth of the signal to be transmitted has to be less than the bandwidth of the channel.
o Ex. Consider a Baseband channel with lower frequency 0Hz and higher frequency 100Hz, hence its bandwidth is 100 (Bandwidth is calculated by getting the difference between the highest and lowest frequency)
o A channel whose bandwidth is more than the bandwidth

of the signal is called Wideband channel
o A channel whose bandwidth is less than the bandwidth of the signal is called Narrowband channel
• Broadband Transmission: Given a bandpass channel, a digital signal cannot be transmitted directly through it
o In broadband transmission we use modulation, i.e. we change the signal to analog signal before transmitting it.

Question sources

Computer Science Exit Exam practice collection · Question 164

Question 223 | NS0334

Which of the following OSI layers is NOT correctly matched to its corresponding data Units (PDU)?

A. Application layer ---> Data

B. Transport layer ---> Segment

C. Physical layer ---> Bit

D. Network layer ---> Frame

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Network-layer PDU is packet, not frame

The PDUs are: application data, transport segment/datagram, network packet, link frame and physical bits. D assigns the link-layer frame to the network layer, making it the mismatch. Transport segment is appropriate for TCP, so B remains a valid conventional example rather than a claim about every transport protocol.

Why each choice is right or wrong

A - Incorrect: Application data is the conventional PDU designation, so this match is valid.

B - Incorrect: TCP uses transport segments; this is a valid conventional transport PDU example.

C - Incorrect: Bits/symbols belong to physical transport, making this match valid.

D - Correct: A frame is link-layer; the network-layer unit is a packet.

Primary note proof

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Question sources

Computer Science Exit Exam · 2018 · Question 63

Question 224 | NS0341

Which of the following layers is responsible for encryption, translation and compression?

A. Presentation layer

B. Data link layer

C. Transport layer

D. Network layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer A: Presentation layer

The three listed operations are representation functions: encoding translation, compression and encryption. NET19 groups them under the OSI presentation role. Link handles frames, transport process delivery and network logical routing; none is the named OSI representation layer. Real implementations may distribute these functions, but the reference-model answer is presentation.

Why each choice is right or wrong

A - Correct: Presentation explicitly covers encoding translation, compression and encryption roles.

B - Incorrect: Data link handles hop frames/access rather than the OSI representation-function grouping.

C - Incorrect: Transport delivers between process endpoints, not the named presentation representation roles.

D - Incorrect: Network handles logical addressing/routing rather than compression/encoding translation.

Primary note proof

Study-note section NET19

> OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Question sources

Computer Science Exit Exam · 2018 · Question 72

Question 225 | NS0352

Assume that you have IPv4 addresses in binary notation 11000001 10000011 00011011 11111111 and you are requested to change into dotted decimal notation. What is the equivalent IP address in decimal dotted notation?

A. 192.174.27.255

B. 192.134.67.254

C. 193.130.28.254

D. 193.131.27.255

Hint

Sum the eight bit weights in each octet independently, then preserve their order in dotted-decimal notation.

Answer D: 193.131.27.255

Using octet weights 128,64,32,16,8,4,2,1 gives 11000001=193, 10000011=131, 00011011=27 and 11111111=255. Combining the four decimal octets gives 193.131.27.255. This is a notation conversion, not a claim about whether that address is assignable as a host under an unspecified prefix.

Why each choice is right or wrong

A - Incorrect: The first octet should be 193 and the second 131, not 192 and 174.

B - Incorrect: All four printed octets differ from the binary-derived values 193,131,27,255.

C - Incorrect: Only the first octet matches; the correct remaining values are 131,27,255.

D - Correct: Each decimal octet matches its eight-bit sum: 193, 131, 27 and 255.

Primary note proof

Study-note section NET22

> Write IPv4 as four decimal octets 0-255, or as 32 binary bits. Use weights 128,64,32,16,8,4,2,1 within each octet. Thus 11000001 10000011 00011011 11111111 is 193.131.27.255. A value such as 257 is impossible. Syntactic validity differs from assignability: an address ending .0 or .255 can be a host under some prefixes.

Question sources

Computer Science Exit Exam · 2018 · Question 89

Question 226 | NS0573

A company has 500 computers distributed across 10 floors of a single building. Which network type BEST describes this setup?

A. WAN

B. MAN

C. LAN

D. Internet

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: LAN

All computers are in one building, the geographic scope of a LAN in NET02. The count of 500 devices or ten floors does not itself make a MAN/WAN; the note explicitly separates device count from geographic category. No metropolitan or distant-region interconnection is supplied.

Why each choice is right or wrong

A - Incorrect: WAN concerns distant regions, not simply many devices on floors of one building.

B - Incorrect: MAN requires metropolitan scope rather than a single-building deployment.

C - Correct: A single building falls within the LAN scope defined by NET02.

D - Incorrect: The Internet is the global network of networks, not the building network described.

Primary note proof

Study-note section NET02

> PAN connects personal devices, such as Bluetooth accessories. LAN covers a local building or campus; MAN spans a metropolitan region; WAN connects distant regions. Device count alone does not determine the category. The Internet is a global network of networks, commonly the largest example of a WAN.

Question sources

Data Communication & Networking review questions · Question 1

Question 227 | NS0574

In which OSI layer does the MAC address operate?

A. Network Layer (Layer 3)

B. Physical Layer (Layer 1)

C. Data Link Layer (Layer 2)

D. Transport Layer (Layer 4)

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: Data link, layer 2

MAC addressing identifies link-layer frame endpoints and forwarding scope. NET13 assigns medium-dependent addressing/access to MAC in the data-link layer. Physical transports signals, network uses logical IP prefixes and transport identifies process endpoints by ports. The word physical in physical address must not be confused with a layer-1 functional assignment.

Why each choice is right or wrong

A - Incorrect: Network-layer addressing uses logical IP addresses rather than the link MAC address.

B - Incorrect: Physical transports signals; the conventional physical-address nickname does not move MAC addressing to layer 1.

C - Correct: MAC addressing and frame delivery are data-link functions at layer 2.

D - Incorrect: Transport identifies process endpoints by ports rather than MAC frame addresses.

Primary note proof

Study-note section NET13

> Link-layer LLC identifies or multiplexes upper-layer protocols and may supply link control; MAC handles medium-dependent addressing/access and frame delivery. Network addressing belongs to layer 3. Ethernet, PPP and HDLC are link protocols. Frame headers may carry addresses and control/type fields; synchronization and exact framing depend on the protocol.

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Study-note section NET16

> IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Question sources

Data Communication & Networking review questions · Question 2

Question 228 | NS0575

Which of the following correctly describes attenuation in data transmission?

A. The delay caused by different frequencies traveling at different speeds

B. The loss of signal strength as it travels over a distance

C. Interference from adjacent communication channels

D. The maximum data rate of a transmission medium

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer B: Loss of signal strength

NET06 defines attenuation as reduced signal strength. Different delays among frequency components describe distortion, adjacent-channel leakage describes crosstalk and maximum rate concerns capacity. B gives the strength-loss mechanism rather than those separate impairments or performance measures.

Why each choice is right or wrong

A - Incorrect: Different component delays change waveform shape, describing distortion rather than attenuation.

B - Correct: A reduction in signal strength during propagation is attenuation.

C - Incorrect: Adjacent-channel interference is crosstalk, a separate noise source.

D - Incorrect: Maximum data rate is channel capacity, not signal-strength loss.

Primary note proof

Study-note section NET06

> Attenuation reduces signal strength. Distortion changes shape, often because frequencies have different delays. Noise adds unwanted energy: thermal noise comes from random thermal motion, crosstalk leaks from nearby channels, and impulse noise is a short disturbance. Amplifiers amplify noise too; digital repeaters regenerate decisions but cannot reconstruct every corrupted bit.

Study-note section NET05

> For an ideal noiseless channel, Nyquist gives $R \leq 2B \log_2(M)$. For Gaussian noise, Shannon gives $C = B \log_2(1+S/N)$, with a linear power ratio; $SNR(dB) = 10 \log_{10}(S/N)$. Bandwidth, signal levels and noise appear explicitly. Distance affects attenuation, delay and often achievable rate, but is not a separate variable in these simplified capacity formulas.

Question sources

Data Communication & Networking review questions · Question 3

Question 229 | NS0576

A network administrator wants to divide the Class C network 192.168.10.0 into 6 subnets. What is the minimum number of bits that must be borrowed from the host portion?

A. 2 bits

B. 3 bits

C. 4 bits

D. 6 bits

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: Borrow 3 bits to obtain at least six equal subnets

For equal-sized subnets of a /24 parent, borrowing b bits yields 2^b blocks. Two bits give only four; three give eight, the minimum power-of-two count that accommodates six required subnets. The resulting mask is /27 with 30 conventional hosts per block. Exactly six equal blocks would not fill the parent; two blocks remain unused.

Why each choice is right or wrong

A - Incorrect: Two borrowed bits yield only four blocks, fewer than the six required.

B - Correct: Three borrowed bits yield eight blocks and are the minimum satisfying at least six.

C - Incorrect: Four bits yield sixteen blocks and are not the minimum required.

D - Incorrect: Six bits yield sixty-four blocks, substantially more than the minimum.

Primary note proof

Study-note section NET26

> To split parent /a into equal /p blocks, borrow b=p-a bits and obtain 2^b subnets. Do not subtract two subnets under modern practice. For a Class B /16 parent split to /19, b=3 gives eight subnets, each with 8192 addresses and 8190 conventional hosts. A /19 alone does not state its parent.

Study-note section NET26

> For H hosts, choose the smallest h satisfying 2^h-2≥H, then p=32-h. For 24 hosts use /27; 52 or 57 use /26; 100 use /25; 500 use /23. For at least six equal subnets choose b=3. A Class C /24 split to /27 yields eight subnets. Always check both the number of subnets and the host capacity.

Question sources

Data Communication & Networking review questions · Question 4

Question 230 | NS0577

Which topology provides the HIGHEST level of redundancy?

A. Star

B. Ring

C. Bus

D. Mesh

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer D: Full mesh among the conventional topology choices

A full mesh connects every pair of nodes, offering multiple alternate paths compared with the basic star, single ring and shared bus. NET08 explains the additional links/interfaces and improved fault tolerance. D assumes the conventional full-mesh interpretation; actual protected rings or partial mesh designs would require more detail for a quantitative comparison.

Why each choice is right or wrong

A - Incorrect: A basic star depends on its central device and lacks the full mesh's many alternate paths.

B - Incorrect: A basic single ring is vulnerable to a break, unlike a full mesh's multiple paths.

C - Incorrect: A shared bus depends on one backbone rather than redundant pairwise paths.

D - Correct: A full mesh supplies pairwise links and multiple alternate paths, at increased cable/interface cost.

Primary note proof

Study-note section NET08

> Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Study-note section NET08

> A full undirected mesh of n nodes needs n(n-1)/2 links and n-1 interfaces per node. For 50 nodes: 50×49/2=1225 links. Redundant paths improve fault tolerance at greater cable and interface cost.

Question sources

Data Communication & Networking review questions · Question 5

Question 231 | NS0578

What is the broadcast address of the network 192.168.17.0/24?

A. 192.168.17.0

B. 192.168.17.254

C. 192.168.17.255

D. 192.168.16.255

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: 192.168.17.255

A /24 prefix fixes the first three octets and leaves eight host bits. Setting them all to one gives final octet 255, so the broadcast is 192.168.17.255. The .0 address is the network boundary and .254 is the last ordinary host; 192.168.16.255 belongs outside the specified /24.

Why each choice is right or wrong

A - Incorrect: 192.168.17.0 has all host bits zero and identifies the network, not broadcast.

B - Incorrect: 192.168.17.254 is the last ordinary host, one below broadcast.

C - Correct: The final eight host bits are all one, giving the /24 broadcast .255.

D - Incorrect: The third octet 16 lies outside network 192.168.17.0/24.

Primary note proof

Study-note section NET24

> Compute network=address AND mask. Host bits zero identify the block start; host bits one give broadcast. With 172.16.28.252/20, the third octet is grouped in blocks of 16, so the network is 172.16.16.0 and broadcast 172.16.31.255. A written host/prefix such as 172.105.180.0/21 need not be an aligned block label: masking yields 172.105.176.0/21.

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Question sources

Data Communication & Networking review questions · Question 6

Question 232 | NS0579

Which device operates at Layer 3 of the OSI model and uses IP addresses to forward data?

A. Hub

B. Bridge

C. Switch

D. Router

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: Router

IP destination-prefix forwarding between networks is the router function in NET11. Hubs repeat signals and bridges/ordinary switches forward MAC frames at layer 2. Some marketed switches include routing, but the unqualified ordinary switch function in this option set is layer 2; router is the explicit matching device.

Why each choice is right or wrong

A - Incorrect: A hub repeats layer-1 signals rather than forwarding on IP prefixes.

B - Incorrect: A bridge makes layer-2 frame-forwarding decisions using MAC addresses.

C - Incorrect: An ordinary switch is layer 2; a specifically layer-3 switch would require the missing routing qualifier.

D - Correct: Routers inspect destination IP prefixes and choose next hops at layer 3.

Primary note proof

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET09

> A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor

eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Question sources

Data Communication & Networking review questions · Question 7

Question 233 | NS0580

UDP is described as a 'connectionless protocol' because it:

A. Cannot transmit video data

B. Does not establish a dedicated connection before sending data and provides no delivery guarantee

C. Only transmits in simplex mode

D. Operates exclusively at the Network Layer

Hint

Identify the requested service or delivery guarantee, then distinguish application roles, transport behavior and address configuration.

Answer B: No setup and no built-in delivery guarantee

UDP's connectionlessness means independent datagrams without built-in setup, not an inability to carry video or an enforced physical duplex mode. It is a transport protocol. B correctly combines no connection establishment with absence of built-in reliable delivery; applications may add their own recovery mechanisms.

Why each choice is right or wrong

A - Incorrect: UDP can carry live media; connectionlessness does not forbid video data.

B - Correct: UDP sends independent datagrams without setup or built-in delivery guarantees.

C - Incorrect: UDP does not impose simplex physical transmission; transport service and physical duplex mode are distinct.

D - Incorrect: UDP is transport-layer, not exclusively a network-layer protocol.

Primary note proof

Study-note section NET17

TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET06

Simplex is one way, half duplex alternates directions, and full duplex permits simultaneous sending/receiving. A walkie-talkie is half duplex; a telephone conversation uses full duplex. Serial sends sequential bits; parallel uses multiple conductors and may suffer skew. Character-oriented asynchronous serial adds start/stop bits; synchronous framing uses ongoing clock recovery. Neither is synonymous with browser asynchronous requests.

Question sources

Data Communication & Networking review questions · Question 8

Question 234 | NS0581

Assume historical classful addressing and an ordinary broadcast network. A host has address 185.28.17.9. How many usable host addresses are in its default classful network?

A. 254

B. 65,534

C. 16,777,214

D. 65,536

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer B: 65,534

The first octet 185 lies in the historical Class B range, whose default prefix is /16. That leaves sixteen host bits, yielding 2^16 = 65,536 total addresses. Reserving the network and directed-broadcast addresses gives 65,534 usable hosts. The explicit classful assumption permits deriving the default prefix; modern CIDR would require a stated prefix instead of inferring it from the address.

Why each choice is right or wrong

A - Incorrect: 254 is the usable count for /24, not the Class B default /16.

B - Correct: Class B defaults to /16, giving 2^16−2 = 65,534 usable hosts.

C - Incorrect: 16,777,214 belongs to /8 rather than the Class B default.

D - Incorrect: 65,536 is the /16 total before network and broadcast reservations.

Primary note proof

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet’s increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Question sources

Data Communication & Networking review questions · Question 9

Question 235 | NS0582

Which transmission mode allows simultaneous bidirectional communication?

A. Simplex

B. Half-Duplex

C. Full-Duplex

D. Multipoint

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer C: Full duplex

The decisive condition is simultaneous communication in both directions. NET06 defines full duplex exactly that way. Half duplex permits both directions at different times and simplex only one. Multipoint describes sharing a medium among endpoints, not whether two endpoints can send and receive at the same instant.

Why each choice is right or wrong

A - Incorrect: Simplex permits communication in one direction only.

B - Incorrect: Half duplex alternates directions and does not permit simultaneous transmission both ways.

C - Correct: Full duplex supports sending and receiving simultaneously.

D - Incorrect: Multipoint describes medium sharing among endpoints rather than a bidirectional timing mode.

Primary note proof

Study-note section NET06

> Simplex is one way, half duplex alternates directions, and full duplex permits simultaneous sending/receiving. A walkie-talkie is half duplex; a telephone conversation uses full duplex. Serial sends sequential bits; parallel uses multiple conductors and may suffer skew. Character-oriented asynchronous serial adds start/stop bits; synchronous framing uses ongoing clock recovery. Neither is synonymous with browser asynchronous requests.

Study-note section NET08

> Physical topology describes link layout; logical topology describes communication/access relationships. Point-to-point links join two endpoints; multipoint links share a medium. A full mesh consists of point-to-point links.

Question sources

Data Communication & Networking review questions · Question 10

Question 236 | NS0583

The OSI Presentation Layer is primarily responsible for:

A. Routing packets between networks

B. Establishing and ending communication sessions

C. Data compression, encryption, and format translation

D. Physical transmission of raw bits

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: Compression, encryption and representation translation

NET19 assigns these three representation functions to the OSI presentation layer. Routing belongs to network, dialogs to session and raw bit signaling to physical. Real software can combine roles, but within the stated OSI model C is the specific matching responsibility set.

Why each choice is right or wrong

A - Incorrect: Routing between networks is the network-layer responsibility.

B - Incorrect: Session setup/ending belongs to the session role rather than presentation representation handling.

C - Correct: Compression, encryption and format translation are the presentation roles explicitly listed in NET19.

D - Incorrect: Physical signaling transports bits/symbols rather than translating data representations.

Primary note proof

Study-note section NET19

> OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET30

> Routing computes paths; forwarding chooses a next hop for an individual packet. Select the matching route with the longest prefix, then apply the routing system’s tie-breaking criteria. A default /0 matches when nothing more specific does. Static routes are configured; dynamic protocols exchange reachability. Distance vector uses neighbor estimates, while link state distributes link information and computes shortest paths. Interior protocols operate within an autonomous system; BGP exchanges routes between systems.

Study-note section NET12

Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Question sources

Data Communication & Networking review questions · Question 11

Question 237 | NS0584

Which statements are not valid universal guarantees for fiber-optic cable? Select all that apply. Interpret B as a capability with suitable optics and installation, and D as a claim of automatic security.

A. Immunity to electromagnetic interference

B. High bandwidth and long transmission distance

C. Lower initial cost than copper cables

D. High data security due to difficulty of tapping

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer C, D: Lower initial cost than copper cables; High data security due to difficulty of tapping

Fiber guides light, resists electromagnetic interference and can support high capacity over long distances when suitable optics and installation are used. Material alone does not determine a universal installed-cost advantage over copper. Nor does physical tapping difficulty automatically ensure data security: fiber can still be tapped. The selected statements are therefore the unsupported universal guarantees about cost and security, rather than the supported transmission capabilities.

Why each choice is right or wrong

A - Incorrect: EMI resistance follows from light-guided transmission, so this is not a rejected guarantee under the stated scope.

B - Incorrect: With suitable optics and installation, high capacity and long reach are supported capabilities, so this is not selected.

C - Correct: Initial cost depends on optics, reach and installation; lower cost than copper is not universal.

D - Correct: Fiber can be tapped, so automatic high security does not follow from the medium.

Primary note proof

Study-note section NET07

> Twisted pairs reduce coupled interference; UTP has no added shield and STP adds shielding. Coax has an inner conductor and outer shield sharing an axis. Fiber guides light and resists electromagnetic interference; it supports high capacity and long reach, but requires suitable optics and installation. Exact speed, cost and reach depend on the specification, not the material alone. Fiber can still be tapped.

Question sources

Data Communication & Networking review questions · Question 13

Question 238 | NS0585

Which option best defines a dedicated server by its primary workload?

A. A server that also acts as a client

B. A machine whose primary workload is supplying services, even though it may use DNS, backup or update services as a client.

C. A server that handles only file storage

D. Any computer in a peer-to-peer network

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B: A machine whose primary workload is supplying services, even though it may use DNS, backup or update services as a client.

A dedicated server concentrates on supplying services, rather than requiring an absolute prohibition on making any client requests. It may still request DNS resolution, backups or software updates. Merely acting as a client does not define dedication, and dedicated workloads are not limited to file storage. The choice identifies the defining workload while allowing ordinary supporting client activity.

Why each choice is right or wrong

A - Incorrect: A dedicated server may act as a DNS/backup client, but that incidental capability is not its defining dedicated workload.

B - Correct: Concentration on service workloads is the defining dedicated-server characteristic.

C - Incorrect: Dedicated service workloads need not be limited to file storage.

D - Incorrect: An arbitrary peer computer is not thereby a dedicated service-workload server.

Primary note proof

Study-note section NET02

> An intranet serves an organization internally; an extranet exposes selected services to approved outsiders. Neither term alone specifies geographic size. A client requests a service; a server supplies it. Peer systems may perform both roles. Dedicated servers concentrate on service workloads but can still be clients of DNS, backup or update services. A ten-user cutoff is a teaching guideline, not a technical law. Centralization simplifies administration but creates dependencies; peer networks still need security.

Question sources

Data Communication & Networking review questions · Question 14

Question 239 | NS0586

Which protocol operates at the Application Layer of the TCP/IP model and is responsible for translating domain names to IP addresses?

A. SMTP

B. FTP

C. HTTP

D. DNS

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer D: DNS

DNS is the name-resolution application protocol in NET20. A/AAAA records provide address data for names. SMTP submits/relays mail, FTP transfers files and HTTP serves web requests; none provides the requested general domain-name record service. Its lookup role is separate from DHCP address assignment.

Why each choice is right or wrong

A - Incorrect: SMTP submits and relays mail rather than resolving hostnames to addresses.

B - Incorrect: FTP transfers files and does not supply the name-resolution record service.

C - Incorrect: HTTP defines web exchanges, not the general hostname-to-address service.

D - Correct: DNS resolves names into records such as A and AAAA address records.

Primary note proof

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Question sources

Data Communication & Networking review questions · Question 15

Question 240 | NS0587

Crosstalk is best described as:

A. Loss of signal strength over distance

B. Signals from one communication channel leaking into an adjacent channel

C. Random electron motion causing electrical noise

D. Delay caused by varying signal frequencies

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer B: Leakage from an adjacent communication channel

Crosstalk is unwanted coupling from a neighboring communication channel, directly matching B. Strength loss is attenuation, random thermal motion creates thermal noise and differential frequency delay produces distortion. These impairments can coexist but their definitions are distinct, so the question's mechanism identifies crosstalk specifically.

Why each choice is right or wrong

A - Incorrect: Signal-strength loss is attenuation rather than coupling from a neighboring channel.

B - Correct: Signals leaking into an adjacent channel are crosstalk.

C - Incorrect: Random electron/thermal motion describes thermal noise.

D - Incorrect: Different component-frequency delays distort waveform shape rather than defining crosstalk.

Primary note proof

Study-note section NET06

> Attenuation reduces signal strength. Distortion changes shape, often because frequencies have different delays. Noise adds unwanted energy: thermal noise comes from random thermal motion, crosstalk leaks from nearby channels, and impulse noise is a short disturbance. Amplifiers amplify noise too; digital repeaters regenerate decisions but cannot reconstruct every corrupted bit.

Question sources

Data Communication & Networking review questions · Question 16

Question 241 | NS0588

How many usable host addresses are available in the subnet 192.168.2.64/26?

A. 32

B. 64

C. 62

D. 30

Hint

Convert the mask to a prefix, separate network and host bits, and apply the relevant subnet-count or usable-host rule.

Answer C: 62 usable hosts

A /26 leaves six host bits, so the block has 64 addresses. In the specified block, .64 is the network and .127 the broadcast; usable hosts span .65-.126 inclusive, totaling 62. The question asks usable hosts, not block size, and /31 point-to-point exceptions do not apply.

Why each choice is right or wrong

A - Incorrect: Thirty-two is not the usable count of this /26 block; /26 has 64 total addresses.

B - Incorrect: Sixty-four is the total block size before network/broadcast reservations.

C - Correct: Six host bits give 64 total addresses minus two reservations, leaving 62 usable hosts.

D - Incorrect: Thirty usable hosts is the /27 result, not /26.

Primary note proof

Study-note section NET25

> For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet's increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Study-note section NET25

> 192.168.2.64/26 spans .64-.127; usable .65-.126 gives 62 hosts. 192.168.10.25/28 spans .16-.31, so its network is .16. 172.17.128.0/21 spans 172.17.128.0-172.17.135.255; .128.255 and .135.0 are valid interior hosts, but .135.255 is broadcast. /27 gives 32 total and 30 usable, not 32 usable.

Question sources

Data Communication & Networking review questions · Question 17

Question 242 | NS0589

Which OSI layer is responsible for establishing, maintaining, and ending communication sessions?

A. Transport Layer

B. Session Layer

C. Application Layer

D. Presentation Layer

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer B: Session layer

The stem explicitly specifies communication sessions, matching the OSI session role rather than merely a TCP connection. NET19 assigns dialog management and checkpoints to that layer. Transport serves process endpoints, application supplies services and presentation handles representations. B is therefore the specific reference-model lifecycle role.

Why each choice is right or wrong

A - Incorrect: Transport connection state is distinct from the explicitly named OSI session/dialog lifecycle.

B - Correct: Session manages dialogs and their setup, maintenance and recovery/end lifecycle.

C - Incorrect: Application supplies user/application network services rather than the distinct session-layer lifecycle.

D - Incorrect: Presentation translates/compresses/encrypts representations instead of managing dialogs.

Primary note proof

Study-note section NET19

> OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Study-note section NET18

> TCP establishes state with SYN, SYN+ACK, then ACK. SYN consumes one sequence position even with no application bytes. An acknowledgment states the next expected sequence number. For client ISN=395 and server ISN=260: client sends SYN seq395; server sends SYN+ACK seq260 ack396; client sends ACK seq396 ack261. Later byte counts advance sequence numbers. FIN also consumes one sequence position.

Question sources

Data Communication & Networking review questions · Question 18

Question 243 | NS0590

A bus topology network has 20 nodes. If the main cable breaks, what happens?

A. Only the two nodes nearest the break are affected

B. The network continues to function normally

C. The entire network goes down

D. Only the node that caused the break is disconnected

Hint

Draw the links and identify shared components, alternate paths and the effect of the specified failure or growth requirement.

Answer C: A main-backbone break can disrupt the whole basic bus segment

In the conventional single-backbone bus assumed by these options, a main-cable break disrupts the shared segment, affecting the network rather than just one endpoint. NET08 qualifies the physical mechanism as an unterminated break. Redundant segmentation or protected designs are not specified, so C expresses the standard basic-bus failure consequence.

Why each choice is right or wrong

A - Incorrect: A main-backbone fault affects shared signaling, not merely the two closest nodes.

B - Incorrect: A broken unterminated backbone does not preserve normal basic-bus operation.

C - Correct: The shared basic bus segment can fail as a whole when its main backbone is broken.

D - Incorrect: A backbone break differs from an isolated endpoint spur failure and need not be limited to one node.

Primary note proof

Study-note section NET08

> Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Question sources

Data Communication & Networking review questions · Question 19

Question 244 | NS0591

Synchronous data transmission differs from asynchronous in that synchronous transmission:

A. Uses start and stop bits for each data chunk

B. Sends data continuously with sender and receiver using the same clock

C. Is always slower than asynchronous

D. Cannot transmit binary data

Hint

Use the cited note definitions to identify the requested function or scope, then check each alternative against that same rule.

Answer B: Ongoing synchronized clock recovery rather than per-character start/stop

NET06 contrasts character start/stop framing with ongoing clock recovery in synchronous transmission. B is the intended synchronized-transfer description; “same clock” should mean synchronized timing, not necessarily a single physical clock device. The note gives no universal speed ordering, and both methods can transmit digital/binary data.

Why each choice is right or wrong

A - Incorrect: Per-character start/stop bits describe character-oriented asynchronous serial transmission.

B - Correct: Synchronous transfer maintains coordinated timing through ongoing clock recovery rather than character start/stop framing.

C - Incorrect: No universal slower-than-asynchronous rule follows from the timing method.

D - Incorrect: Synchronous framing can carry binary data; the timing convention does not prohibit it.

Primary note proof

Study-note section NET06

> Simplex is one way, half duplex alternates directions, and full duplex permits simultaneous sending/receiving. A walkie-talkie is half duplex; a telephone conversation uses full duplex. Serial sends sequential bits; parallel uses multiple conductors and may suffer skew. Character-oriented asynchronous serial adds start/stop bits; synchronous framing uses ongoing clock recovery. Neither is synonymous with browser asynchronous requests.

Question sources

Data Communication & Networking review questions · Question 20

Question 245 | NS0592

What is the PRIMARY function of the Network Layer (Layer 3)?

A. Providing physical transmission of bits

B. Error detection and MAC addressing

C. End-to-end routing of data between different networks using IP addresses

D. Establishing communication sessions between applications

Hint

Match the requested scope or data unit to the layer duties in the note; distinguish hop delivery from end-to-end process delivery.

Answer C: Logical IP addressing and routing across networks

NET12 gives network-layer logical addressing and routing; NET11 explains forwarding IP packets among networks by destination prefixes. C directly states that responsibility. The other options assign physical signaling, link MAC/error detection and session-dialog management, which belong to different model layers.

Why each choice is right or wrong

A - Incorrect: Transporting raw physical bits/symbols is the physical-layer function.

B - Incorrect: MAC addressing and hop frame error detection are data-link responsibilities.

C - Correct: Network-layer logical addressing and routing carry IP packets between networks.

D - Incorrect: Establishing application dialogs is a session-layer role rather than IP routing.

Primary note proof

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Study-note section NET19

> OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations

distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Question sources

Data Communication & Networking review questions · Question 21

Question 246 | NS0593

Which of the following IPv4 addresses is INVALID?

A. 192.168.0.1

B. 10.0.0.255

C. 172.16.254.1

D. 10.256.0.1

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer D: 10.256.0.1 has an impossible octet

An IPv4 decimal octet must fit eight bits, from 0 through 255. D has 256 and is therefore syntactically invalid. A-C each meet that range condition. B's .255 ending alone does not establish a broadcast without a prefix, so it cannot be rejected merely by its last octet.

Why each choice is right or wrong

A - Incorrect: All four octets are within 0-255, so this notation is valid.

B - Incorrect: Every octet is valid; a .255 ending alone does not determine broadcast status without a prefix.

C - Incorrect: All four values fit eight-bit octets and form valid dotted-decimal notation.

D - Correct: The second octet 256 exceeds the maximum representable eight-bit value 255.

Primary note proof

Study-note section NET22

Write IPv4 as four decimal octets 0-255, or as 32 binary bits. Use weights 128,64,32,16,8,4,2,1 within each octet. Thus 11000001 10000011 00011011 11111111 is 193.131.27.255. A value such as 257 is impossible. Syntactic validity differs from assignability: an address ending .0 or .255 can be a host under some prefixes.

Question sources

Data Communication & Networking review questions · Question 22

Question 247 | NS0594

In IPv6, addresses are how many bits long?

A. 32 bits

B. 64 bits

C. 96 bits

D. 128 bits

Hint

Separate address width, syntax, reserved scope and the supplied prefix or historical classful convention.

Answer D: 128 bits

NET23 explicitly gives IPv6 addresses a 128-bit width. IPv4 uses 32 bits; 64 may describe a common subnet interface boundary but is not the whole IPv6 address length. Ninety-six bits is likewise not the full width. D therefore answers the requested address size.

Why each choice is right or wrong

A - Incorrect: Thirty-two bits is the IPv4 address width, not IPv6.

B - Incorrect: Sixty-four bits is not the complete IPv6 address width.

C - Incorrect: Ninety-six bits does not match the specified 128-bit IPv6 address length.

D - Correct: IPv6 addresses contain 128 bits as explicitly stated in NET23.

Primary note proof

Study-note section NET23

> IPv6 has 128-bit hexadecimal addresses, flexible prefixes and no IPv4-style broadcast. Sources, not routers, fragment IPv6 packets. IPv6 does not automatically encrypt traffic or make authentication unnecessary. Compare IPv6's larger 128-bit address space and encryption support separately; specify the property being evaluated.

Study-note section NET16

> IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Study-note section NET-S06

> A common IPv6 unicast subnet uses a 64-bit subnet prefix and a 64-bit interface identifier. This split does not make the complete address 64 bits: it remains 128 bits. Special uses and explicitly different prefix lengths must be considered separately.

Question sources

Data Communication & Networking review questions · Question 23

Question 248 | NS0595

A bridge decides whether to forward a frame to another network segment by examining the:

A. IP address in the packet header

B. Port number in the segment header

C. Source and destination MAC addresses

D. TTL (Time to Live) field

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer C: Learn source MACs and forward according to destination MAC

A bridge learns which sources are reachable through each port and consults the destination MAC when forwarding/filtering a frame. C names the relevant link addresses. IP, transport ports and TTL belong to higher-layer packet/segment processing, not the ordinary bridge forwarding decision. Unknown destinations may be flooded, so learning does not imply always-one-port forwarding.

Why each choice is right or wrong

A - Incorrect: Destination IP prefixes drive routing, not ordinary layer-2 bridge frame forwarding.

B - Incorrect: Transport ports identify process endpoints and are not the bridge MAC-table lookup key.

C - Correct: Source MAC informs learning and destination MAC informs forwarding/filtering to the appropriate segment.

D - Incorrect: TTL is an IP header field used in routed packet lifetime, not ordinary bridging.

Primary note proof

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET17

> Ports identify transport endpoints; the application can choose a transport and request socket buffer sizes, subject to OS limits. A TCP connection is identified by source/destination IP and source/destination port, with TCP understood. IPv4 protocol numbers identify upper protocols; 80 is a usual HTTP port, not an IP protocol number. Clients often use ephemeral ports. SCTP is a reliable message-oriented transport with multiple streams.

Study-note section NET16

IP supplies connectionless best-effort packet delivery, not end-to-end ordering or retransmission. IPv4 addresses are 32 bits. Its minimum header is 20 bytes and can grow to 60 bytes; it is not a 32-bit header. IHL counts 32-bit words; total length includes header and payload. TTL limits lifetime across routers; the protocol field identifies TCP, UDP or another upper-layer protocol.

Question sources

Data Communication & Networking review questions · Question 24

Question 249 | NS0596

Circuit switching guarantees bandwidth for the duration of a call. What is the main DISADVANTAGE of this?

A. High latency during transmission

B. Bandwidth is wasted when no data is actively being transmitted

C. Packets may arrive out of order

D. It cannot support voice communication

Hint

Compare signal repetition, frame forwarding, packet routing and policy enforcement using the stated device or traffic assumptions.

Answer B: Reserved bandwidth can be wasted during silence

Circuit switching holds resources for the call even while endpoints are silent, so unused reservation cannot be statistically shared as packet traffic would be. B gives that cost. Circuit switching does not inherently deliver packets out of order or prevent voice transmission; traditional telephone calls are a standard example.

Why each choice is right or wrong

A - Incorrect: Reserved circuits can offer predictable transfer behavior; setup cost is distinct from universally high transfer latency.

B - Correct: Capacity remains reserved during silence and can be wasted when no data is sent.

C - Incorrect: Independent out-of-order packet routing is not the defining reserved-circuit disadvantage.

D - Incorrect: Traditional telephone calls are the note's standard circuit-switching example, so voice is supported.

Primary note proof

Study-note section NET31

> Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Question sources

Data Communication & Networking review questions · Question 25

Question 250 | NS0597

Under the textbook direct line-of-sight wireless-link convention, which option gives the most complete set of listed technologies whose direct path can be blocked by buildings or terrain? Treat radio waves as the broad category rather than assuming every radio propagation mode needs line of sight.

A. Radio waves

B. Microwaves

C. Infrared

D. Both B and C

Hint

Compare the signal mechanism, medium properties or transmission direction against the definitions and stated physical assumptions.

Answer D: Microwave and infrared under the conventional line-of-sight assumption

Conventional directional microwave links need suitable path visibility, and infrared generally does not penetrate walls. Obstacles can therefore interrupt the direct links specified here. Both individual technologies fit, so the combined option is the complete response. Radio waves is a broader physical category that includes microwave but also other propagation modes; the question does not assert that all radio links share the same line-of-sight restriction.

Why each choice is right or wrong

A - Incorrect: The broad radio category includes propagation modes beyond the specified line-of-sight comparison.

B - Incorrect: Microwave fits the stated direct-link case, but this alone omits infrared.

C - Incorrect: Infrared fits the stated obstructed-direct-path case, but this alone omits microwave.

D - Correct: Both microwave and infrared fit the specified comparison, making this the complete set.

Primary note proof

Study-note section NET07

> Radio, microwave and infrared are unguided media. Ground waves follow Earth, sky waves can reflect from ionized layers, and line-of-sight paths require suitable visibility. Microwaves are radio waves in the broader physical classification. Infrared generally does not penetrate walls. Directionality depends on antennas: omnidirectional means broad azimuth coverage, not equal radiation in every 3-D direction. Legacy Ethernet straight-through cables join unlike MDI/MDI-X roles; crossover joins like roles, such as PC-PC or switch-switch. Auto-MDI-X can remove that requirement.

Question sources

Data Communication & Networking review questions · Question 26

Question 251 | NS0598

What does CIDR stand for, and what problem did it solve?

A. Classless Inter-Domain Routing; it solved address wastage in classful addressing

B. Classful Internet Domain Routing; it solved packet loss

C. Centralized IP Domain Routing; it solved routing loops

D. Classless Internet Data Routing; it solved latency in WANs

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer A: Classless Inter-Domain Routing; flexible prefixes reduce rigid class allocation waste

CIDR replaces rigid class allocation. Flexible prefixes allow blocks sized to requirements and aggregation of aligned blocks, addressing waste from fixed historical class sizes. The other expansions are incorrect and their proposed packet-loss, loop or latency cures are not the defined purpose.

Why each choice is right or wrong

A - Correct: This gives the correct expansion and relates flexible prefix allocation to avoiding rigid class-size waste.

B - Incorrect: Classful Internet Domain Routing is the wrong expansion, and CIDR is not defined as a packet-loss cure.

C - Incorrect: Centralized IP Domain Routing is not the acronym expansion or the stated mechanism.

D - Incorrect: Classless Internet Data Routing is incorrect; CIDR is not defined as a general WAN-latency cure.

Primary note proof

Study-note section NET28

> CIDR, Classless Inter-Domain Routing, replaces rigid class allocation and supports aggregation. Combine contiguous aligned blocks sharing leading bits; unrelated adjacent-looking labels may not aggregate cleanly. Four consecutive /26 blocks covering a /24 summarize to /24. More-specific routes must remain available where the aggregate would otherwise send traffic to the wrong next hop.

Study-note section NET26

> For H hosts, choose the smallest h satisfying 2^h-2≥H, then p=32-h. For 24 hosts use /27; 52 or 57 use /26; 100 use /25; 500 use /23. For at least six equal subnets choose b=3. A Class C /24 split to /27 yields eight subnets. Always check both the number of subnets and the host capacity.

Question sources

Data Communication & Networking review questions · Question 27

Question 252 | NS0599

How many subnets are created when you borrow 5 bits from the host portion of a Class B network?

A. 5

B. 16

C. 32

D. 64

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: 32 subnets

Borrowing five bits yields 2^5=32 equal subnets under modern counting. The parent Class B /16 would become /21, but the direct subnet count only needs the specified five borrowed bits. NET26 says not to subtract two subnets under modern practice.

Why each choice is right or wrong

A - Incorrect: Five is the number of borrowed bits, not the exponential number of resulting subnet combinations.

B - Incorrect: Sixteen subnets would use four borrowed bits rather than five.

C - Correct: Five borrowed bits have 32 distinct combinations and therefore produce 32 subnets.

D - Incorrect: Sixty-four subnets would require six borrowed bits.

Primary note proof

Study-note section NET26

> To split parent /a into equal /p blocks, borrow b=p-a bits and obtain 2^b subnets. Do not subtract two subnets under modern practice. For a Class B /16 parent split to /19, b=3 gives eight subnets, each with 8192 addresses and 8190 conventional hosts. A /19 alone does not state its parent.

Question sources

Data Communication & Networking review questions · Question 28

Question 253 | NS0600

The 'forwarding table' in a bridge is also known as the:

A. Routing table

B. ARP table

C. MAC table

D. IP table

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: MAC table

The bridge forwarding database records MAC-to-port associations learned from source addresses and used for destination lookups. MAC table is therefore the matching name. A routing table maps IP prefixes to next hops; ARP associates on-link IPv4 addresses with link addresses. Those are different tables even if one device maintains several of them.

Why each choice is right or wrong

A - Incorrect: Routing tables describe destination IP prefixes and next hops rather than bridge MAC-to-port mappings.

B - Incorrect: ARP caches associate IPv4 next hops with link addresses, not MAC addresses with bridge ports.

C - Correct: The bridge's learned source-MAC-to-port forwarding database is its MAC table.

D - Incorrect: IP table does not name the layer-2 learned MAC forwarding database described in NET10.

Primary note proof

Study-note section NET-S06

> A bridge or switch's MAC address table is its learned forwarding database: it associates MAC addresses with ports for frame forwarding. It differs from an ARP cache, which associates an IPv4 next hop with a link address, and from a routing table, which selects next hops for IP prefixes.

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Question sources

Data Communication & Networking review questions · Question 29

Question 254 | NS0601

Which layer of the OSI model is responsible for defining voltage levels, cable specifications, and how bits are physically transmitted?

A. Data Link Layer

B. Network Layer

C. Physical Layer

D. Transport Layer

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: Physical layer

Voltage, cable/interface characteristics and signal transport are the physical-layer scope in NET09. Data-link functions operate on frames and MAC access, network handles IP routing and transport handles process delivery. The wording explicitly asks how bits are physically represented and carried, so C is the corresponding layer.

Why each choice is right or wrong

A - Incorrect: Data link manages frames/access rather than defining the physical voltage and cable interface.

B - Incorrect: Network manages logical addressing/routing rather than electrical signal specifications.

C - Correct: Physical-layer specifications include connectors, signaling, voltage and timing.

D - Incorrect: Transport supplies process delivery rather than cable and voltage definitions.

Primary note proof

Study-note section NET09

> A NIC supplies a computer’s network interface and commonly implements both physical and link functions. Calling a NIC a physical interface does not mean all its functions are layer 1. Physical specifications include connectors, signaling, voltage, timing and physical coding/modulation; the PHY interface to MAC handles hardware signaling.

Study-note section NET12

> Sending proceeds from application data to transport segment/datagram, network packet, link frame and physical bits/symbols; receiving reverses the order. Each layer serves the layer above. A router removes a received link frame, inspects IP, and creates a new link frame for the next hop. MAC addresses usually change across routed hops; end-to-end IP addresses ordinarily remain until translation.

Question sources

Data Communication & Networking review questions · Question 30

Question 255 | NS0602

In a star topology, what happens if the central hub fails?

A. Only the hub's nearest neighbor is affected

B. The entire network fails

C. The network continues operating in bus mode

D. Nodes automatically reconnect in ring topology

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer B: The dependent star loses connectivity when its center fails

In a basic star every endpoint depends on the central hub for communication with other endpoints. Failure of that common center therefore affects the whole star. No alternate center or automatic topology reconfiguration is specified; the network does not become a bus or ring merely because the hub stops functioning.

Why each choice is right or wrong

A - Incorrect: The center serves all star endpoints, so its failure is not limited to the nearest neighbor.

B - Correct: All endpoints depend on the central hub, so the basic star loses inter-endpoint connectivity.

C - Incorrect: A hub failure does not physically or logically rewire the star into an operational bus.

D - Incorrect: No automatic ring reconnection mechanism follows from the basic star topology.

Primary note proof

Study-note section NET08

> Bus uses a shared backbone; an unterminated break can disrupt the segment. Star connects endpoints through a central device: one edge failure isolates that endpoint, whereas central failure affects the star. Ring forms a loop; basic single rings are vulnerable to a break, while protected dual rings differ. Token passing is associated with Token Ring but is an access method, not a necessary property of every ring. Tree is hierarchical; hybrid combines arrangements.

Question sources

Data Communication & Networking review questions · Question 31

Question 256 | NS0603

What type of noise CANNOT be completely eliminated because it is caused by the random motion of electrons due to temperature?

A. Crosstalk

B. Impulse noise

C. Thermal noise

D. Intermodulation noise

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: Thermal noise

The stated cause-random motion associated with temperature-is thermal noise in NET06. Crosstalk is neighboring-channel coupling and impulse noise is a short disturbance. D names a different noise phenomenon not defined by random thermal motion. The cause identifies C without requiring a claim that all other real noise can be perfectly eliminated.

Why each choice is right or wrong

A - Incorrect: Crosstalk arises from coupling between nearby communication channels, not the stated thermal-motion cause.

B - Incorrect: Impulse noise is a short disturbance rather than persistent random thermal motion.

C - Correct: Thermal noise is explicitly caused by random thermal motion, matching the temperature-dependent clue.

D - Incorrect: This option does not match the random thermal-motion definition supplied in NET06.

Primary note proof

Study-note section NET06

> Attenuation reduces signal strength. Distortion changes shape, often because frequencies have different delays. Noise adds unwanted energy: thermal noise comes from random thermal motion, crosstalk leaks from nearby channels, and impulse noise is a short disturbance. Amplifiers amplify noise too; digital repeaters regenerate decisions but cannot reconstruct every corrupted bit.

Study-note section NET-S07

> Intermodulation distortion occurs when multiple signals interact through a nonlinear component and generate unwanted combination-frequency products. Its mechanism differs from random thermal motion, adjacent-channel leakage and an isolated impulse disturbance.

Question sources

Data Communication & Networking review questions · Question 32

Question 257 | NS0604

Which IP address class has a first-octet range of 128-191?

A. Class A

B. Class B

C. Class C

D. Class D

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer B: Class B

NET22 gives first-octet 128-191 for historical Class B, corresponding to leading bits 10. Ordinary Class A is 1-126, Class C is 192-223 and Class D uses leading 1110 for multicast. Since the question explicitly asks classes, the historical categorization is appropriate rather than inferring a current CIDR prefix.

Why each choice is right or wrong

A - Incorrect: Ordinary historical Class A uses first octets 1-126 rather than 128-191.

B - Correct: Historical Class B spans first octets 128-191, with leading bits 10.

C - Incorrect: Historical Class C spans 192-223, above the supplied interval.

D - Incorrect: Class D multicast begins with 1110, outside the Class B 128-191 interval.

Primary note proof

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Question sources

Data Communication & Networking review questions · Question 33

Question 258 | NS0605

The Session Layer's 'checkpointing' feature means that it:

A. Encrypts data to protect it from interception

B. Compresses data to reduce transmission time

C. Saves the state of a communication session at regular intervals so it can be resumed after interruption

D. Translates data between different character encoding formats

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: Checkpoint state supports resuming after interruption

A checkpoint records recoverable progress/state so interrupted work can resume from an agreed point. NET19 assigns such coordination to the session role. Encryption, compression and encoding translation belong to representation handling and do not by themselves preserve resumable dialog progress. C matches the checkpoint recovery purpose.

Why each choice is right or wrong

A - Incorrect: Encryption conceals content and does not define saving resumable session progress.

B - Incorrect: Compression reduces representation size rather than recording recovery checkpoints.

C - Correct: Saving agreed progress/state enables continuation after interruption rather than restarting from the beginning.

D - Incorrect: Encoding translation changes representation format, not checkpoint recovery state.

Primary note proof

Study-note section NET19

> OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Question sources

Data Communication & Networking review questions · Question 34

Question 259 | NS0606

Which of the following is a De Jure standard?

A. A manufacturer's proprietary cable format adopted industry-wide

B. A standard officially approved by ISO or IEEE

C. A standard developed by a single corporation without regulatory approval

D. A protocol widely used but never formally documented

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer B: A formally approved standard

NET03 defines de jure by formal approval. B directly states official approval, while industry adoption without that approval is de facto. Corporate origin does not alone decide status, but C specifically lacks formal approval and D emphasizes informal use. Therefore B is the option establishing the defining approval condition.

Why each choice is right or wrong

A - Incorrect: Widespread adoption alone demonstrates de facto use rather than formal de jure approval.

B - Correct: Official approval matches the defining de jure criterion in NET03.

C - Incorrect: A corporate format without the stated formal approval does not meet the offered de jure definition.

D - Incorrect: Widespread informal use without formal approval is de facto rather than de jure.

Primary note proof

Study-note section NET03

> Protocol syntax defines message structure, semantics defines field meanings, and timing defines when/how fast exchanges occur. A standard supports interoperability. De facto means established through use; de jure means formally approved. ISO develops international standards; IEEE develops LAN standards such as 802.3 and 802.11; ITU-T develops telecommunications standards. ANSI coordinates US standards, EIA is associated with interface standards, and FCC is a regulator.

Question sources

Data Communication & Networking review questions · Question 35

Question 260 | NS0607

Parallel data transmission sends multiple bits simultaneously. Its key DISADVANTAGE compared to serial transmission is:

A. It is always slower

B. It cannot be used for digital data

C. It requires more transmission lines and can suffer from interference between lines

D. It only works over fiber-optic cable

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: More conductors and interline impairments

Parallel transfer needs several conductors to carry simultaneous bits. Those paths can experience skew and neighboring-channel coupling, as NET06's impairment definitions explain. C states the extra-line and interference burden. No universal slower-speed claim, inability to carry digital data or fiber-only restriction follows from parallel transmission.

Why each choice is right or wrong

A - Incorrect: Parallel is not universally slower; performance depends on implementation, distance and synchronization.

B - Incorrect: Parallel transmission can carry digital bits simultaneously, exactly as the stem states.

C - Correct: Multiple conductors add wiring and can suffer skew or coupled interference between adjacent lines.

D - Incorrect: Parallel transmission is not restricted to fiber and commonly concerns multiple electrical conductors.

Primary note proof

Study-note section NET06

> Simplex is one way, half duplex alternates directions, and full duplex permits simultaneous sending/receiving. A walkie-talkie is half duplex; a telephone conversation uses full duplex. Serial sends sequential bits; parallel uses multiple conductors and may suffer skew. Character-oriented asynchronous serial adds start/stop bits; synchronous framing uses ongoing clock recovery. Neither is synonymous with browser asynchronous requests.

Study-note section NET06

> Attenuation reduces signal strength. Distortion changes shape, often because frequencies have different delays. Noise adds unwanted energy: thermal noise comes from random thermal motion, crosstalk leaks from nearby channels, and impulse noise is a short disturbance. Amplifiers amplify noise too; digital repeaters regenerate decisions but cannot reconstruct every corrupted bit.

Question sources

Data Communication & Networking review questions · Question 36

Question 261 | NS0608

A packet-switching network may be UNSUITABLE for real-time voice communication because:

A. It requires a dedicated circuit

B. Variable latency and possible packet loss can disrupt real-time audio

C. It cannot carry voice signals

D. All bandwidth is reserved, leaving none for voice

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer B: Variable delay and packet loss can disturb real-time audio

Real-time audio is sensitive to late or missing data. Packet networks may introduce variable queueing delay and loss, so B gives a plausible suitability issue. Modern voice over packets remains possible with buffering, scheduling and capacity planning; it is not intrinsically prohibited or dependent on reserved circuits.

Why each choice is right or wrong

A - Incorrect: Datagram packet switching does not require a dedicated circuit before every communication.

B - Correct: Variable queueing latency and loss can make audio arrive late or incomplete.

C - Incorrect: NET31 explicitly allows voice over packets with suitable engineering.

D - Incorrect: Statistical link sharing differs from reserving all bandwidth for established circuits.

Primary note proof

Study-note section NET31

> Circuit switching sets up a path with reserved resources, transfers data, then releases it. Predictable allocated bandwidth trades against setup/blocking and wasted reservation during silence. A traditional telephone call is the standard example. Packet switching divides data into packets that share links statistically; queueing can add variable delay and loss. This does not make modern voice-over-packet impossible: buffering, scheduling and capacity planning manage it.

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in

setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Question sources

Data Communication & Networking review questions · Question 37

Question 262 | NS0609

For a Class C network subnetted to /27, how many usable hosts per subnet are available?

A. 32

B. 30

C. 28

D. 62

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer B: 30 usable hosts

A /27 leaves five host bits regardless of the historical class of its parent. Those bits provide 32 total addresses. On an ordinary broadcast subnet, network and broadcast reserve two, leaving 30 usable host addresses. Parent classification matters for subnet counting, not this explicit-prefix host calculation.

Why each choice is right or wrong

A - Incorrect: Thirty-two is the total address count before reserving network and broadcast.

B - Correct: Five host bits give 32−2=30 ordinary usable hosts.

C - Incorrect: Twenty-eight does not follow from the stated five host bits and two standard reservations.

D - Incorrect: Sixty-two usable hosts corresponds to /26 rather than the supplied /27.

Primary note proof

Study-note section NET25

For ordinary broadcast LANs let h=32-p. Total addresses=2^h; usable hosts=2^h-2, reserving network and directed broadcast. Find the changing octet’s increment as 256-mask octet, round the address down to a multiple, and extend through the next boundary minus one.

Question sources

Data Communication & Networking review questions · Question 38

Question 263 | NS0610

Which routing type sends data to the NEAREST of multiple hosts sharing the same IP address?

A. Multicast

B. Broadcast

C. Unicast

D. Anycast

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer D: Anycast

Anycast routes to one member of a set sharing the address, typically the nearest by routing metrics. That is not necessarily geographic distance. Multicast targets a subscribed group, broadcast all in scope and ordinary unicast a single designated destination. D uniquely matches the shared-address nearest-member rule.

Why each choice is right or wrong

A - Incorrect: Multicast targets a subscribed group rather than selecting only one nearest shared-address member.

B - Incorrect: Broadcast targets all destinations within its scope, not the nearest member.

C - Incorrect: Ordinary unicast targets one designated destination without the shared-address nearest-member semantics.

D - Correct: Anycast selects one shared-address member, normally nearest according to routing metrics.

Primary note proof

Study-note section NET30

> Flooding sends copies on eligible outgoing links with duplicate/lifetime controls; random walk picks a neighbor probabilistically. Unicast targets one destination, broadcast all in a scope, multicast a subscribed group and anycast one member sharing an address, normally nearest under routing metrics. Multi-destination packets can list multiple targets; this is not necessarily identical to broadcast. Group multicast trees use mechanisms such as reverse-path checks; Ethernet spanning tree is a different layer-2 mechanism.

Question sources

Data Communication & Networking review questions · Question 39

Question 264 | NS0611

The term 'coaxial' refers to the fact that the cable's inner conductor and outer shield:

A. Are made of the same conductive material

B. Share the same central axis

C. Carry signals in opposite directions

D. Are both covered in plastic insulation

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer B: The conductor and shield share an axis

The name coaxial refers to a common geometric axis for the inner conductor and surrounding shield. NET07 states this directly. Neither identical material, opposite signal directions nor a particular insulation covering defines the term; those are separate construction or usage details rather than the shared-axis relationship.

Why each choice is right or wrong

A - Incorrect: The definition concerns geometric arrangement, not whether both conductive parts use the same material.

B - Correct: The inner conductor and outer shield share a central axis, giving the coaxial name.

C - Incorrect: Opposite signal directions are not the defining common-axis geometry.

D - Incorrect: Insulation details do not define coaxial; sharing an axis does.

Primary note proof

Study-note section NET07

> Twisted pairs reduce coupled interference; UTP has no added shield and STP adds shielding. Coax has an inner conductor and outer shield sharing an axis. Fiber guides light and resists electromagnetic interference; it supports high capacity and long reach, but requires suitable optics and installation. Exact speed, cost and reach depend on the specification, not the material alone. Fiber can still be tapped.

Question sources

Data Communication & Networking review questions · Question 40

Question 265 | NS0612

Which option best describes a protocol-translating gateway joining unlike networked systems?

A. A device that boosts signal strength over long distances

B. A device that connects networks using the same protocol

C. A device that connects two or more networks using different protocols, topologies, or architectures

D. A device that filters traffic based on MAC addresses

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: A device that connects two or more networks using different protocols, topologies, or architectures

Gateway has several networking meanings. A default IP gateway is commonly a router, while a protocol gateway translates between unlike protocols or application formats. The question asks for that translating role, so the option joining unlike systems fits. Signal regeneration belongs to repeaters, same-protocol routing does not require translation, and MAC-based filtering describes bridging or switching rather than protocol conversion.

Why each choice is right or wrong

A - Incorrect: Signal regeneration/strength extension is a repeater function rather than either gateway meaning defined here.

B - Incorrect: Connecting networks using the same protocol describes routing without the required protocol translation.

C - Correct: A protocol gateway joins unlike systems through the translation required by the stated scope.

D - Incorrect: MAC-address frame filtering describes bridging/switching rather than the gateway distinction in NET11.

Primary note proof

Study-note section NET11

> Routers forward IP packets between networks using destination prefixes and next hops. They separate broadcast domains and normally do not forward local link broadcasts. A default gateway is usually a router, while a protocol gateway can translate application formats; therefore a gateway is not inherently a device that implements every OSI layer.

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does

not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET09

A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Question sources

Data Communication & Networking review questions · Question 41

Question 266 | NS0613

An organization is assigned the Class A IP address 10.200.240.4. What is the network address?

A. 10.0.0.0

B. 10.200.0.0

C. 10.200.240.0

D. 0.0.0.0

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer A: 10.0.0.0 under the explicitly requested Class A default

Because the question explicitly assigns Class A, its historical default /8 supplies the intended network/host boundary. Keep first octet 10 and zero the remaining 24 host bits to obtain 10.0.0.0. If a different CIDR prefix were supplied it would override the default, but none is given in this class-specific question.

Why each choice is right or wrong

A - Correct: The Class A /8 default keeps first octet 10 and clears the remaining three octets.

B - Incorrect: This assumes a /16 boundary instead of the explicitly invoked Class A /8 default.

C - Incorrect: This assumes /24, retaining host bits that the Class A default would clear.

D - Incorrect: Clearing the network octet too loses the assigned Class A network identifier 10.

Primary note proof

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Study-note section NET24

> Compute network=address AND mask. Host bits zero identify the block start; host bits one give broadcast. With 172.16.28.252/20, the third octet is grouped in blocks of 16, so the network is 172.16.16.0 and broadcast 172.16.31.255. A written host/prefix such as 172.105.180.0/21 need not be an aligned block label: masking yields 172.105.176.0/21.

Question sources

Data Communication & Networking review questions · Question 42

Question 267 | NS0614

UTP cable is most susceptible to which type of interference?

A. Thermal noise only

B. Electromagnetic interference (EMI) and crosstalk

C. Infrared interference

D. Gamma radiation

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer B: EMI and crosstalk

UTP is unshielded copper: twisting reduces coupled interference but does not remove all electromagnetic coupling. NET06 identifies crosstalk as leakage from nearby channels. B names the relevant ordinary interference types, whereas thermal-only excludes other sources and infrared/gamma are not the note's cabling-susceptibility mechanism.

Why each choice is right or wrong

A - Incorrect: Thermal noise is not the only impairment; unshielded pairs can experience coupled electromagnetic interference.

B - Correct: EMI and adjacent-pair/channel crosstalk are relevant to unshielded copper, though twisting reduces coupling.

C - Incorrect: Infrared is an optical/unguided signaling category rather than the stated ordinary copper-coupling issue.

D - Incorrect: Gamma radiation is not the interference mechanism identified for UTP in the supplied note.

Primary note proof

Study-note section NET07

> Twisted pairs reduce coupled interference; UTP has no added shield and STP adds shielding. Coax has an inner conductor and outer shield sharing an axis. Fiber guides light and resists electromagnetic interference; it supports high capacity and long reach, but requires suitable optics and installation. Exact speed, cost and reach depend on the specification, not the material alone. Fiber can still be tapped.

Study-note section NET06

> Attenuation reduces signal strength. Distortion changes shape, often because frequencies have different delays. Noise adds unwanted energy: thermal noise comes from random thermal motion, crosstalk leaks from nearby channels, and impulse noise is a short disturbance. Amplifiers amplify noise too; digital repeaters regenerate decisions but cannot reconstruct every corrupted bit.

Question sources

Data Communication & Networking review questions · Question 43

Question 268 | NS0615

For a known unicast whose destination is on a different learned switch port in the same VLAN, what is the main forwarding difference between an active hub and an ordinary Ethernet switch?

A. Hubs operate at Layer 3; switches operate at Layer 2

B. Hubs send data to all connected devices; switches send data only to the intended device

C. Hubs use IP addresses; switches use MAC addresses

D. Hubs support fiber-optic; switches do not

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer B: Hubs send data to all connected devices; switches send data only to the intended device

An active hub repeats incoming signals onto its other ports without examining addresses. A switch learns MAC-to-port mappings and, for the specified known unicast on another port, forwards the frame only to that destination port. Unknown unicasts and broadcasts are different cases and can be flooded. The scope restriction makes the intended contrast accurate without treating selective forwarding as a universal rule for all switch traffic.

Why each choice is right or wrong

A - Incorrect: Hubs are layer-1 repeaters, not layer-3 routers; ordinary switches are layer 2.

B - Correct: In this known-unicast case the hub repeats to other ports, while the switch sends only to the learned destination port.

C - Incorrect: Hubs do not use IP addresses to decide forwarding; they repeat signals without address inspection.

D - Incorrect: Hub versus switch is a signal-repetition versus frame-forwarding distinction; it does not impose the claimed prohibition on optical switch links.

Primary note proof

Study-note section NET10

> A bridge or ordinary Ethernet switch operates at layer 2. It learns source-MAC-to-port mappings, looks up the destination, forwards a known unicast to its port, filters it if the destination is on the incoming segment, and floods an unknown unicast or broadcast within its VLAN. A switch does not always send only to one destination port. Wireless access points normally bridge wireless and wired link-layer traffic.

Study-note section NET09

A repeater regenerates layer-1 signals without inspecting destination addresses. An active hub is a multiport repeater: it repeats incoming signals onto other ports, placing half-duplex endpoints in one collision domain. A passive hub is a connector. A hub neither learns MAC addresses nor eliminates collisions. An endpoint such as a network printer consumes services; routers, hubs and access points are intermediary devices.

Study-note section NET-S07

Ethernet switches can have fiber-optic interfaces with suitable optical transceivers. A forwarding role therefore does not impose a copper-only restriction: distinguish what the device does with frames from the physical medium used by its ports.

Question sources

Data Communication & Networking review questions · Question 44

Question 269 | NS0616

In multicast routing, data is sent to:

A. All devices on the network regardless of interest

B. A single specific destination

C. The nearest host among multiple hosts sharing an address

D. Only the devices that have expressed interest in receiving the data

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer D: A subscribed/interested receiver group

NET30 distinguishes multicast group delivery from unicast, broadcast and anycast. Receivers express membership/interest in a group, rather than every network device receiving by definition or only one nearest destination being selected. D matches that destination-set meaning. Delivery still depends on the deployed network's forwarding and membership mechanisms.

Why each choice is right or wrong

A - Incorrect: All devices regardless of interest describes broadcast scope rather than multicast group selection.

B - Incorrect: One specific destination is ordinary unicast rather than a receiver group.

C - Incorrect: One nearest shared-address member is anycast.

D - Correct: Multicast targets a subscribed group of receivers that expressed interest.

Primary note proof

Study-note section NET30

> Flooding sends copies on eligible outgoing links with duplicate/lifetime controls; random walk picks a neighbor probabilistically. Unicast targets one destination, broadcast all in a scope, multicast a subscribed group and anycast one member sharing an address, normally nearest under routing metrics. Multi-destination packets can list multiple targets; this is not necessarily identical to broadcast. Group multicast trees use mechanisms such as reverse-path checks; Ethernet spanning tree is a different layer-2 mechanism.

Question sources

Data Communication & Networking review questions · Question 45

Question 270 | NS0617

What IPv4 address class is designed for multicast addressing?

A. Class A

B. Class B

C. Class C

D. Class D

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer D: Class D

Historical IPv4 Class D begins with bits 1110 and is designated for multicast. Classes A, B and C describe historical unicast network allocations, with differing default prefixes. The question explicitly asks that historical class designation, so D is supported independently of how modern unicast prefixes are allocated.

Why each choice is right or wrong

A - Incorrect: Class A is historical unicast network space rather than the multicast class.

B - Incorrect: Class B is historical unicast space with a /16 default, not multicast.

C - Incorrect: Class C is historical unicast space with a /24 default.

D - Correct: Class D, identified by leading 1110, is designated for IPv4 multicast.

Primary note proof

Study-note section NET22

> Historical classes use leading patterns: A 0 with /8, B 10 with /16, C 110 with /24; D 1110 is multicast and E 1111 reserved. Ordinary class-A first octets are 1-126 because 0 and 127 have special uses. B spans 128-191 and C 192-223. Default usable counts are 2^24-2, 2^16-2 and 254. Modern CIDR requires the actual prefix and overrides class defaults.

Question sources

Data Communication & Networking review questions · Question 46

Question 271 | NS0618

Which OSI layer provides sequencing, acknowledgments and receiver flow control for end-to-end reliable process delivery, as in TCP?

A. Network Layer

B. Data Link Layer

C. Transport Layer

D. Session Layer

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: Transport Layer

TCP uses sequence numbers, acknowledgments and windows to provide reliable ordered delivery between transport endpoints. Reliable link protocols can use similar mechanisms, but their scope is a single hop. The explicit end-to-end process requirement therefore selects transport. IP supplies best-effort network delivery, while session coordinates dialogs and checkpoints rather than defining TCP's ordered-byte-stream reliability and receiver flow control.

Why each choice is right or wrong

A - Incorrect: IP is best effort and does not supply the listed reliable acknowledgment/retransmission service.

B - Incorrect: Data-link reliability is hop-local rather than the required end-to-end process service.

C - Correct: Transport supplies TCP's end-to-end sequencing, acknowledgments and receiver flow control.

D - Incorrect: Session manages dialogs/checkpoints rather than the specifically listed link/TCP reliability machinery.

Primary note proof

Study-note section NET17

> TCP offers a reliable ordered byte stream using sequence numbers, acknowledgments, retransmission and windows. Flow control protects the receiver; congestion control protects network capacity. UDP sends independent datagrams with ports and a checksum but no built-in setup, delivery guarantee, retransmission or ordering. Live interactive traffic may choose UDP when late data is less useful than lost data. Application protocols can add reliability over UDP.

Study-note section NET14

> Controlled access uses polling, reservations or a token to grant turns; correctly coordinated access avoids contention collisions. Channelization divides frequency, time or code resources. Stop-and-wait sends one frame and waits for ACK, retransmitting on timeout; sequence bits distinguish duplicates. Sliding windows permit multiple unacknowledged frames. Link reliability/flow control is hop-local; transport reliability/flow control is end-to-end. Ethernet detects corruption but does not itself guarantee reliable delivery.

Study-note section NET19

> OSI session roles establish/manage dialogs and can coordinate checkpoints so transfer resumes after interruption. Presentation roles translate encodings and handle compression/encryption. Compression reduces representation size; encryption hides content and need not shrink it. Application protocols provide services directly to applications. Practical implementations

distribute these functions differently; TLS is not proof of a separate physical presentation-layer device.

Question sources

Data Communication & Networking review questions · Question 47

Question 272 | NS0619

For the IP address 192.168.2.64/26, what is the broadcast address?

A. 192.168.2.65

B. 192.168.2.126

C. 192.168.2.127

D. 192.168.2.255

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: 192.168.2.127

A /26 has blocks of 64 addresses. Starting at .64, this block runs through .127; the final address has all six host bits one and is broadcast. .65 and .126 are ordinary first/last hosts. .255 belongs to another /26 block, so it is not this subnet's broadcast.

Why each choice is right or wrong

A - Incorrect: 192.168.2.65 is the first usable host, not the all-host-bits-one address.

B - Incorrect: 192.168.2.126 is the last usable host, one address below broadcast.

C - Correct: The .64-.127 /26 block ends at .127, its directed broadcast.

D - Incorrect: The .255 address is outside this .64-.127 subnet block.

Primary note proof

Study-note section NET25

> 192.168.2.64/26 spans .64-.127; usable .65-.126 gives 62 hosts. 192.168.10.25/28 spans .16-.31, so its network is .16. 172.17.128.0/21 spans 172.17.128.0-172.17.135.255; .128.255 and .135.0 are valid interior hosts, but .135.255 is broadcast. /27 gives 32 total and 30 usable, not 32 usable.

Study-note section NET24

> Compute network=address AND mask. Host bits zero identify the block start; host bits one give broadcast. With 172.16.28.252/20, the third octet is grouped in blocks of 16, so the network is 172.16.16.0 and broadcast 172.16.31.255. A written host/prefix such as 172.105.180.0/21 need not be an aligned block label: masking yields 172.105.176.0/21.

Question sources

Data Communication & Networking review questions · Question 48

Question 273 | NS0620

Which of the following CORRECTLY describes the role of SMTP?

A. Translates domain names to IP addresses

B. Governs how web pages are requested and delivered

C. Governs how email messages are sent from a client to a mail server

D. Enables file transfer between devices over the internet

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: SMTP submits and relays email

NET21 assigns SMTP mail submission and relay. A client sending a message to a mail server is therefore a correct SMTP role. DNS resolves names, HTTP serves web requests and FTP transfers files. This does not assign end-user mailbox retrieval to SMTP; POP3/IMAP serve that separate task.

Why each choice is right or wrong

A - Incorrect: DNS resolves domain names to address records rather than SMTP.

B - Incorrect: HTTP governs web requests/responses, not SMTP.

C - Correct: SMTP submits mail from a client to a server and relays messages between mail servers.

D - Incorrect: File transfer is the FTP service role in this comparison, not SMTP's defining mail role.

Primary note proof

Study-note section NET21

> HTTP defines web request/response semantics; it is stateless, and a message body may be absent. HTTPS adds TLS protection. FTP transfers files using control and separate data connections; it is not the same protocol as SFTP. SMTP submits and relays mail; POP3 retrieves messages and IMAP accesses/synchronizes server mailboxes. Saying SMTP handles both sending and end-user mailbox retrieval is inaccurate.

Study-note section NET20

> DNS resolves names to records, including IPv4 A and IPv6 AAAA addresses; it can use UDP or TCP port 53. DHCP dynamically leases IPv4 settings, commonly address, mask, gateway and DNS server. Its basic exchange is discover, offer, request, acknowledge. IP itself does not assign addresses.

Question sources

Data Communication & Networking review questions · Question 49

Question 274 | NS0621

An omnidirectional antenna transmits signals in which pattern?

A. A narrow beam toward a specific target

B. Only vertically upward

C. In all directions simultaneously, creating 360-degree coverage

D. In a focused line-of-sight path only

Hint

Apply the cited definitions to the stated scope; for calculations, derive the result from the supplied prefix, bit count or resource rule.

Answer C: 360-degree azimuth coverage, with a nonuniform vertical pattern

NET07 defines omnidirectional as broad azimuth coverage. C is correct when its 360 degrees refers to the horizontal plane, not equal power in every three-dimensional direction. Directional beams or upward-only radiation do not describe this pattern. Antenna elevation patterns can still have lobes/nulls, so the answer retains that scope qualification.

Why each choice is right or wrong

A - Incorrect: A narrow targeted beam is directional rather than omnidirectional azimuth coverage.

B - Incorrect: Upward-only radiation does not provide the broad around-the-antenna azimuth pattern.

C - Correct: 360-degree azimuth coverage is the intended meaning, without asserting equal power in all three-dimensional directions.

D - Incorrect: A single focused path describes a directional antenna pattern.

Primary note proof

Study-note section NET07

> Radio, microwave and infrared are unguided media. Ground waves follow Earth, sky waves can reflect from ionized layers, and line-of-sight paths require suitable visibility. Microwaves are radio waves in the broader physical classification. Infrared generally does not penetrate walls. Directionality depends on antennas: omnidirectional means broad azimuth coverage, not equal radiation in every 3-D direction. Legacy Ethernet straight-through cables join unlike MDI/MDI-X roles; crossover joins like roles, such as PC-PC or switch-switch. Auto-MDI-X can remove that requirement.

Question sources

Data Communication & Networking review questions · Question 50

Data Structure and Algorithms

Questions & Worked Answers

271 questions + 4 source variants

Question 1 | Q00012

When we use a hash table data structure, our hash function should map the expected inputs (keys) as evenly as possible over its output range(indexes). This property of good hash function is known as_____.

A. Uniformity

B. Determinism

C. Continuity

D. Data normalization

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer A: Uniformity

DSA25 defines uniformity as spreading expected keys evenly among addresses. That directly matches the question's wording. Determinism is a different obligation: repeated equal keys receive the same address for the chosen function. Continuity and data normalization do not name this distribution property, so only Uniformity answers the literal stem.

Why each choice is right or wrong

A - Correct: Uniformity describes spreading expected input keys evenly across the output address range, exactly as the stem requires.

B - Incorrect: Determinism requires repeated equal keys to map identically; it does not require an even address distribution.

C - Incorrect: Continuity is not the key-distribution property defined for hash functions in DSA25.

D - Incorrect: Data normalization does not name the even-distribution requirement of a hash function.

Primary note proof

Study-note section DSA25

> Determinism means repeated equal keys get the same address for the chosen function. Uniformity means expected keys spread evenly. Universal hashing chooses one function randomly from a suitable family for the table, independently of a fixed key set; it does not choose a fresh unrelated function for each lookup. Random choice controls collision probability, not a promise of zero collisions.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 12

Question 2 | Q00013

In a one-based array representation of a binary heap, the root is at index 1. If a node at index i has a right child, at which index is that child stored?

A. 2*i+1

B. i/2

C. (i+1)/2

D. 2*i

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. 2*i+1

Heap child formulas depend on whether indexing begins at zero or one. With the explicitly stated one-based convention, the two children of node i occupy positions 2i and 2i+1. The latter is the right child. Integer division by two instead identifies a parent in this convention. The existence condition matters because an index computed beyond the current heap size does not designate an actual child.

Why each choice is right or wrong

A - Correct: 2i+1 is the one-based right-child index.

B - Incorrect: floor(i/2) computes the parent for nonroot nodes.

C - Incorrect: (i+1)/2 does not compute the right child.

D - Incorrect: 2i is the one-based left child.

Primary note proof

Study-note section DSA23

> Zero-based array representation: root index 0, left=2i+1, right=2i+2, parent=floor((i-1)/2) for i>0. One-based representation uses children 2i,2i+1 and parent floor(i/2). A child exists only if its index is inside the current heap size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 23-37

> iii. Heap Sort
> Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
> each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
> Algorithm:

1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 13

Question 3 | Q00014

Use iterative sequential search and iterative binary search on random-access arrays, with constant-size indices. For binary search, reverse comparisons on descending arrays. Compare tight asymptotic auxiliary space and worst-case element probes as n grows. Which statement is false?

A. Binary searching algorithm can work on list of data which is stored in descending order

B. Sequential and binary searching algorithm are not equal in terms of memory efficiency

C. Sequential searching algorithm can work on sorted and unsorted list of data

D. Binary search has a smaller tight worst-case probe order than sequential search on sorted random-access arrays.

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer B: B. Sequential and binary searching algorithm are not equal in terms of memory efficiency

Both iterative algorithms keep only a fixed number of indices and temporary values, giving constant auxiliary space. Their worst-case probe counts differ: sequential search can inspect every element, whereas binary search halves the ordered interval each time. Binary search can also handle descending order with reversed comparisons, and sequential search does not require sorted input. Under these specified implementations, the unequal-memory claim is the false statement.

Why each choice is right or wrong

A - Incorrect: Descending binary search works with reversed comparisons.

B - Correct: Both iterative implementations use constant auxiliary space, so this inequality claim is false.

C - Incorrect: A sequential scan works with either input ordering.

D - Incorrect: Binary search has logarithmic rather than linear worst-case probes.

Primary note proof

Study-note section DSA28

> Compare the target with successive elements until equal or exhausted. Unsorted data is allowed; sorted data is allowed too. A first-element match costs Theta(1). An absent target or last-position match needs n comparisons, giving Theta(n) worst time. Uniformly likely successful positions average (n+1)/2 comparisons, still linear.

Study-note section DSA29

> Worst probes are floor($\log_2 n$)+1 for n>0: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

> Linear/Sequential searching
> • Works with unsorted List
> • Loop through the list starting at the first element
> • Time is proportional to the size of input (n) and we call this time complexity O(n).

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 14

Question 4 | Q00045

For ordinary array sorting implementations, which listed algorithm uses a temporary element buffer proportional to n rather than only a fixed number of saved elements? Count recursive call frames separately from auxiliary element storage.

A. Insertion sort

B. Selection sort

C. Merge sort

D. Quick sort

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: C. Merge sort

Ordinary array mergesort merges two sorted halves through a temporary buffer that can hold a number of elements proportional to n. Insertion sort retains one item while shifting, and selection sort needs fixed temporary storage for a minimum position and an exchange. Ordinary quicksort partitions inside the original array. Its recursive stack is counted separately here, so the linear element-buffer requirement uniquely identifies mergesort.

Why each choice is right or wrong

A - Incorrect: Insertion saves one item and shifts within the same array.

B - Incorrect: Selection retains a minimum position and a fixed-size swap temporary.

C - Correct: Ordinary array merging needs a linear auxiliary element buffer.

D - Incorrect: In-array partitioning needs no full merge buffer; recursive frames are excluded by the stated convention.

Primary note proof

Study-note section DSA37

> Ordinary array merging needs a linear auxiliary buffer.

Study-note section DSA34

> For each remaining item, save its value, shift larger preceding values right, and insert it into the gap.

Study-note section DSA33

> At position i, scan the remaining suffix for its minimum, then exchange it with A[i].

Study-note section DSA36

> Quicksort normally rearranges in the same array but recursive storage can be linear in the worst case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

MOE Computer Science Exit Exam · 2016 · Question 45

Question 5 | Q00066

Which data structure is the most appropriate for sharing a printer in a network environment?

A. Stack

B. Binary search tree

C. Full binary tree

D. Queue

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: Queue

The module explicitly identifies print-server jobs as a queue application. FIFO preserves arrival order among pending jobs. Evidence: assigned PDF pages 8, 6, 9.

Why each choice is right or wrong

A - Incorrect: A stack would give the most recent print request priority. See assigned PDF 8, 6, 9.

B - Incorrect: BST ordering by key is not arrival-order service. See assigned PDF 8, 6, 9.

C - Incorrect: A full tree shape does not specify job service order. See assigned PDF 8, 6, 9.

D - Correct: The print server maintains a queue of print jobs. See assigned PDF 8, 6, 9.

Primary note proof

Study-note section DSA14

> FIFO suits equal-priority printer jobs, shared-resource waiting, asynchronous buffers and level-order exploration. A play sequence can also use a queue. Distinguish airline seat records from waiting requests; an application name alone does not force one representation.

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 41-48

> Application of Queue
> Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
> • Print server- maintains a queue of print jobs
> • Disk Driver- maintains a queue of disk input/output requests
> • Task scheduler in multiprocessing system- maintains priority queues of processes

• Telephone calls in a busy environment -maintains a queue of telephone calls
• Simulation of waiting line- models real life queues (e.g. supermarkets checkout, customer service line (Bank, custom, airport etc))

Data Structure and Algorithms module | PDF page 6 | lines 15-22

"A stack is an ordered list in which all insertions and deletions are made at one end, called the top". Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 66

Question 6 | Q00076

Which one of the following is true about Queue data structure?

A. Queue data structure applies first come first served policy

B. Queue data structure uses last in first out rule

C. Queue data structure does not have efficient basic operations

D. Queue data structure cannot be implemented using linked list

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Queue data structure applies first come first served policy

FIFO serves the first arrival first; the module also explicitly gives linked-list enqueue and dequeue. Therefore A is true and the supplied D key contradicts page 8. Evidence: assigned PDF pages 7, 8.

Why each choice is right or wrong

A - Correct: First come first served is stated under queue applications. See assigned PDF 7, 8, 8.

B - Incorrect: LIFO describes the stack, whereas queue deletion is at the front. See assigned PDF 7, 8, 8.

C - Incorrect: A blanket inefficiency claim is unsupported; front removal and rear insertion can use fixed pointer updates. See assigned PDF 7, 8, 8.

D - Incorrect: The module expressly implements the queue as a linked list. See assigned PDF 7, 8, 8.

Primary note proof

Study-note section DSA14

An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

“A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 8 | lines 35-37

III. Linked list implementation of enqueue and dequeue operations
Enqueue- is inserting a node at the end of a linked list
Dequeue- is deleting the first node in the list

Data Structure and Algorithms module | PDF page 8 | lines 41-48

Application of Queue
Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
• Print server- maintains a queue of print jobs
• Disk Driver- maintains a queue of disk input/output requests
• Task scheduler in multiprocessing system- maintains priority queues of processes
• Telephone calls in a busy environment -maintains a queue of telephone calls
• Simulation of waiting line- models real life queues (e.g. supermarkets checkout, customer service line (Bank, custom, airport etc))

Question sources

MOE Computer Science Exit Exam · 2016 · Question 76

Question 7 | Q00148

An unsorted singly linked list maintains only its head pointer. Which option gives the complete set of listed operations with tight worst-case constant time? There is no cached tail, predecessor index or positional index.

A. Insertion at the front of the linked list

B. Insertion at the end of the linked list

C. Deletion of the front node of the linked list

D. Deletion of the last node of the linked list

E. A&C

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer E: E. A&C

Inserting at the head links the new node to the old head and updates head. Deleting the head advances head and releases one node. Both use constant work. Appending requires traversal to find the tail, and deleting the tail requires its predecessor, also found by traversal. The complete constant-time set is therefore insertion at the front together with deletion of the front, which is option E.

Why each choice is right or wrong

A - Incorrect: Front insertion is constant, but this alone omits the also-constant front deletion.

B - Incorrect: Head-only append traverses the list to find its tail.

C - Incorrect: Front deletion is constant, but this alone omits the also-constant front insertion.

D - Incorrect: Tail deletion must locate its predecessor by traversal.

E - Correct: This combines exactly the two listed constant-time operations.

Primary note proof

Study-note section DSA09

> Prepend with new.next=head; head=new. To insert after a known predecessor p, set new.next=p.next before p.next=new. Appending with only head requires a tail traversal; maintaining tail avoids it. Delete the head by saving old=head, advancing head=head.next and releasing old in C++.

Study-note section DSA09

> Deleting the tail of a singly linked list requires its predecessor, even when tail itself is known. Handle empty and single-node lists separately; after removing the only node, set both head and

tail to null. To remove a known successor, bypass it before deallocation. Clearing an owned n-node chain visits and releases every node.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 5 | lines 13-24

Step 3: Then add the node at the end of the list. For this we need to check whether the list is empty or not. If list is empty the new node will be the first (Start) node, otherwise navigate forward starting form the first node till last node found and add it next to this node which make the new node the last node.
if (start == NULL) // If list is empty, then new node become start node
start = temp;
else {
temp2 = start; // list not empty! Thus temp2 starts from first node
while (temp2->next!= NULL) { // till the current last node found navigates forward
temp2 = temp2->next; // Move to next link in chain
}
temp2->next = temp; // new node (temp) appended at the end of the list

Data Structure and Algorithms module | PDF page 6 | lines 4-13

2. Make temp1 point to whatever the start pointer is pointing to.
3. If the next pointer of what temp1 indicates is NULL, then we've found the last node of the list, so
jump to step 7.
4. Make another pointer, temp2, point to the current node in the list.
5. Make temp1 point to the next item in the list.
6. Go to step 3.
7. If you get this far, then the temporary pointer, temp1, should point to the last item in the list and
the other temporary pointer, temp2, should point to the last-but-one item.

8. Delete the node pointed to by temp1.
9. Mark the next pointer of the node pointed to by temp2 as NULL - it is the new last node.

Question sources

Addis Ababa University mock exam · Question 50

Question 8 | Q00150

Which of the following data structure is more appropriate to represent a heap tree?

A. Two dimensional array

B. Doubly linked list

C. Linear array

D. Single linked list

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Linear array

The heap algorithm directly maps tree nodes to Data indices and uses arithmetic child indices, establishing the linear array representation. Evidence: assigned PDF pages 11.

Why each choice is right or wrong

A - Incorrect: A second array dimension is unnecessary for one index per node. See assigned PDF 11.

B - Incorrect: A doubly linked list lacks the given direct child-index arithmetic. See assigned PDF 11.

C - Correct: Data[0] and children 2*i+1, 2*i+2 use a one-dimensional array. See assigned PDF 11.

D - Incorrect: A singly linked chain does not directly represent the two children by these indices. See assigned PDF 11.

Primary note proof

Study-note section DSA23

Zero-based array representation: root index 0, left=2i+1, right=2i+2, parent=floor((i-1)/2) for i>0. One-based representation uses children 2i,2i+1 and parent floor(i/2). A child exists only if its index is inside the current heap size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Question sources

Addis Ababa University mock exam · Question 52

Question 9 | Q00151

Linear search is highly inefficient compare to binary search when dealing with_____?

A. Small, unsorted arrays

B. Large, sorted arrays

C. Small, sorted arrays

D. Large, unsorted arrays

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: Large, sorted arrays

The comparison n versus log2 n becomes substantial for large n, and binary searching requires ordering. Thus both size and sortedness matter. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: Unsorted input does not satisfy binary-search prerequisites. See assigned PDF 3, 3.

B - Correct: Large sorted arrays provide both the prerequisite and the asymptotic advantage. See assigned PDF 3, 3.

C - Incorrect: Sorting permits binary search, but small size does not produce the large growth-rate difference requested. See assigned PDF 3, 3.

D - Incorrect: Large unsorted arrays still cannot be binary-searched without preprocessing. See assigned PDF 3, 3.

Primary note proof

Study-note section DSA28

> Compare the target with successive elements until equal or exhausted. Unsorted data is allowed; sorted data is allowed too. A first-element match costs Theta(1). An absent target or last-position match needs n comparisons, giving Theta(n) worst time. Uniformly likely successful positions average (n+1)/2 comparisons, still linear.

Study-note section DSA29

> Worst probes are floor($\log_2 n$)+1 for n>0: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

> Linear/Sequential searching
> • Works with unsorted List
> • Loop through the list starting at the first element
> • Time is proportional to the size of input (n) and we call this time complexity O(n).

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Question sources

Addis Ababa University mock exam · Question 53

Question 10 | Q00152

Considering the array elements (independently for this question):
int a[6]={11,13,15,17,19,21}
Implementing stack, the elements after pop(); push(9); are:

A. 9, 11, 13, 15, 17, 19

B. 13, 15, 17, 19, 21, 9

C. 11, 13, 15, 17, 19, 9

D. 9, 13, 15, 17, 19, 21

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer C: 11, 13, 15, 17, 19, 9

Take the listed array in bottom-to-top order, with 21 at top. Pop removes 21; push appends 9 at the same top slot, leaving 11,13,15,17,19,9. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Incorrect: This puts 9 at the bottom and shifts surviving data. See assigned PDF 6, 6.

B - Incorrect: This removes 11, which is a queue-style front removal. See assigned PDF 6, 6.

C - Correct: Only the final top entry changes from 21 to 9. See assigned PDF 6, 6.

D - Incorrect: This replaces the bottom entry 11 rather than the top. See assigned PDF 6, 6.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 6 | lines 24-42

```
I. Array implementation (static memory).
Push(n): Adding item at top
top is the current top of stack and n is its
maximum size
Begin
if (top == n) then stack is full;
else
top := top+1;
stack[top] := item;
end:
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;
```

Question sources

Addis Ababa University mock exam · Question 54

Question 11 | Q00153

A capacity-six circular queue uses all six slots with front=0 and size=6. Its physical array is [11,13,15,17,19,21]. After dequeue() and enqueue(9), what is the logical front-to-rear sequence?

A. 9, 11, 13, 15, 17, 19

B. 13, 15, 17, 19, 21, 9

C. 11, 13, 15, 17, 19, 9

D. Queue overflow

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer B: B. 13, 15, 17, 19, 21, 9

The initial front element is 11, so dequeue removes it and advances front to the slot containing 13. The queue now has five items. Circular insertion reuses the freed physical slot zero for 9 and restores size six. Reading from the new front around to the rear gives 13,15,17,19,21,9. Physical slot order differs from this requested logical service order.

Why each choice is right or wrong

A - Incorrect: This would remove 21 and prepend 9 rather than remove the front.

B - Correct: This is the front-to-rear order after removing 11 and appending 9.

C - Incorrect: This incorrectly retains 11 and removes the rear 21.

D - Incorrect: Circular reuse makes room for the insertion after the removal.

Primary note proof

Study-note section DSA15

> A simple noncircular array queue may report overflow when rear reaches the final index despite released slots at the front. A circular queue reuses slots with modulo arithmetic. Use a count convention: front points to the next removal, size counts items, and insertion slot is (front+size)%C. Empty means size==0; full means size==C.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 7 | lines 18-23

> I. Simple Array implementation (static memory).
> A simple array structure can be used to implement enqueue & dequeue operations. Thus item will
> be added at the end of the array whereas dequeue operation removes items from the front. In simple
> array implementation, to add item it checks whether there is room at the rear or not. Therefore, even though there are free released spaces at the front ot the array the queue implementation might
> generate overflow message. This is the key drawback to use simple array to implement Queue.

Data Structure and Algorithms module | PDF page 7 | lines 24-26

> II. Circular array implementation of Queue Operations
> To overcome the above drawback of simple array implementation a circular array can be used, in
> which freed spaces are re-used to store data.

Question sources

Addis Ababa University mock exam · Question 55

Question 12 | Q00154

A circular queue has capacity six, front=0 and size=6, using physical slots [11,13,15,17,19,21]. It can use all six slots. After dequeue() and enqueue(9), list the physical array in index order 0 through 5.

A. 9, 11, 13, 15, 17, 19

B. 13, 15, 17, 19, 21, 9

C. 11, 13, 15, 17, 19, 9

D. 9, 13, 15, 17, 19, 21

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer D: D. 9, 13, 15, 17, 19, 21

Dequeue reads 11 at index zero, advances front to index one and reduces size to five. The count-based insertion formula then gives (front+size)%6=(1+5)%6=0. Enqueue therefore writes 9 into the freed slot zero. All other physical slots retain their old values. The logical service order begins at 13, but the requested physical index order begins at the newly stored 9.

Why each choice is right or wrong

A - Incorrect: This drops 21 and retains the removed 11.

B - Incorrect: This is logical front-to-rear order, not physical index order.

C - Incorrect: This overwrites 21 instead of reusing slot zero.

D - Correct: Slot zero receives 9 and slots 1-5 remain unchanged.

Primary note proof

Study-note section DSA15

> A simple noncircular array queue may report overflow when rear reaches the final index despite released slots at the front. A circular queue reuses slots with modulo arithmetic. Use a count convention: front points to the next removal, size counts items, and insertion slot is (front+size)%C. Empty means size==0; full means size==C.

Study-note section DSA15

> Dequeue reads A[front], sets front=(front+1)%C and decreases size. An alternative reserves one unused slot and uses next(rear)==front for full; its empty test and capacity differ. State the convention before tracing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 7 | lines 24-26

> II. Circular array implementation of Queue Operations
> To overcome the above drawback of simple array implementation a circular array can be used, in
> which freed spaces are re-used to store data.

Question sources

Addis Ababa University mock exam · Question 56

Question 13 | Q00171

The data structure used in the standard implementation of Breadth First Search is ?

A. Stack

B. Array

C. Linked List

D. Queue

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: Queue

Breadth-first search repeatedly dequeues a vertex and enqueues its undiscovered neighbors. FIFO ordering ensures earlier distance layers are processed before later ones. Arrays or linked nodes can store this frontier, but Queue names its required access discipline. A stack instead serves latest pending work.

Why each choice is right or wrong

A - Incorrect: A stack serves the latest pending vertex first and does not preserve BFS distance-layer order.

B - Incorrect: An array can store a queue, but an array alone specifies no FIFO frontier discipline.

C - Incorrect: Linked nodes can represent a queue, but Linked List alone does not specify breadth-first processing.

D - Correct: A FIFO queue processes discovered vertices by distance layer, the standard BFS frontier rule.

Primary note proof

Study-note section DSA41

Mark the source when enqueuing it. Repeatedly dequeue a vertex, examine its neighbours, and enqueue each unvisited neighbour while marking it. Store predecessor[v] when discovering v if a path must be reconstructed. FIFO ensures vertices at distance d are explored before later distance layers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

“A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Addis Ababa University mock exam · Question 73

Question 14 | Q00214

Suppose we define a static method totalEvenLoc() that returns the total for even-index items of an integer list as shown, on the right, below. What is the Big O running time of the method if lst is a LinkedList of size N?

```
public static int totalEvenLoc(List<Integer> lst) {
int sum = 0;
for(int i = 0; i < lst.size(); i += 2)
sum += lst.get(i);
return sum;
}
```

A. O(N)

B. O(1)

C. O(N^2)

D. O(logN)

E. O(NlogN)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: O(N^2), tightly Θ(N^2)

The loop visits about N/2 even indices, but LinkedList.get(i) walks from the nearer endpoint. A linear number of those indices lie near the middle, each requiring linear traversal. DSA07 analyzes exactly this access pattern and obtains Θ(N^2) total work. Incrementing by two changes a constant factor, not the quadratic order.

Why each choice is right or wrong

A - Incorrect: O(N) would require constant-cost indexed access; LinkedList.get(i) instead traverses nodes.

B - Incorrect: A growing number of accesses and additions prevents a constant bound on total work.

C - Correct: A linear number of middle-index accesses each traverse a linear number of nodes, yielding quadratic time.

D - Incorrect: Neither the iteration count nor the accumulated linked-node traversal has logarithmic growth.

E - Incorrect: Quadratic traversal grows faster than N log N, so this is not a valid upper bound.

Primary note proof

Study-note section DSA07

> Indexed access is not constant for every List implementation. Java LinkedList.get(i) walks from the nearer endpoint. Summing get(0), get(2), get(4), ... over a length-N list still gives Theta(N^2), because a linear number of middle indices each require linear traversal. An iterator scans once in linear time. An exact executable runtime requires valid code and an explicit cost convention.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
> to the chain.

Data Structure and Algorithms module | PDF page 5 | lines 26-34

III. Displaying the list of Nodes
After adding nodes to the list, we need to display the list of nodes on the screen. Here are the steps required to display all nodes item in the list:
1. Set a temporary pointer to point to the same thing as the start pointer.
2. If the pointer points to NULL, display the message "End of list" and stop.
3. Otherwise, display the details of the node pointed to by the start pointer.
4. Make the temporary pointer point to the same thing as the next pointer of the node it is currently indicating.
5. Jump back to step 2.

Data Structure and Algorithms module | PDF page 3 | lines 2-9

I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:
II. Big-Omega Notation (Ω): provides an asymptotic lower bound; f(n)= Ω(g (n)) means that the growth rate of f(n) is greater than or equal to g(n).
III. Theta Notation (Θ): If f(n)= Θ (g(n)), then g(n) is an asymptotically tight bound for f(n) i.e f(n) and g(n) have the same rate of growth.

Question sources

Mock exam 4 · Question 16

Question 15 | Q00215

The array 10, 8, 6, 2, 1, 4, 5 is organized into a full heap (heap and full BT). Which array represents the heap after the first and second biggest elements are removed?

A. 6 5 4 2 1

B. 6 4 5 2 1

C. 6 4 5 1 2

D. 6 5 4 1 2

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: 6 5 4 2 1

Remove 10 by moving 5 to root and adjusting: [8,5,6,2,1,4]. Remove 8 by moving 4 to root and adjusting through child 6: [6,5,4,2,1]. These are standard root-last swaps and downward heap restoration. Evidence: assigned PDF pages 11.

Why each choice is right or wrong

A - Correct: Matches both root deletions and adjustment steps. See assigned PDF 11.

B - Incorrect: A valid max heap, but not the result of these root-last adjustments. See assigned PDF 11.

C - Incorrect: Changes the lower pair 2,1 as well as giving a different adjustment history. See assigned PDF 11.

D - Incorrect: Reverses the unchanged lower pair 2,1. See assigned PDF 11.

Primary note proof

Study-note section DSA24

> Insert at the next complete-tree position and swap upward while parent order is violated. To remove the root, move the final element to root, reduce heap size, then swap downward with the more dominant child until order holds. Each path has logarithmic length. Building bottom-up by sinking internal nodes takes linear total time; repeated independent insertions give an O(n log n) construction bound.

Study-note section DSA24

> Max-heap [10,8,6,2,1,4,5]: remove 10, move 5 to root and sink, giving [8,5,6,2,1,4]. Remove 8, move 4 up then exchange with 6: [6,5,4,2,1].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 23-37

> iii. Heap Sort
> Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
> each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
> Algorithm:
> 1. Construct a binary tree
> • The root node corresponds to Data[0].
> • If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
> 2. Construct the heap tree from initial binary tree using "adjust" process.
> Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Question sources

Mock exam 4 · Question 17

Question 16 | Q00216

Think of a reference implementation of a classical queue with front and rear pointing to the appropriate locations in the queue. Which of these references will change during an insertion into a possibly empty queue?

A. Neither change

B. Only front changes

C. None of the choices is correct

D. Both may change.

E. Only rear changes

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: Both may change.

Insertion into an empty linked queue makes the new node both front and rear. Insertion into a nonempty queue changes rear while front remains. Consequently both references may change across the allowed cases. Evidence: assigned PDF pages 7, 8, 5.

Why each choice is right or wrong

A - Incorrect: Insertion must update at least rear to include the new node. See assigned PDF 7, 8, 5.

B - Incorrect: Rear changes too; front alone is insufficient. See assigned PDF 7, 8, 5.

C - Incorrect: D describes the empty-queue case correctly. See assigned PDF 7, 8, 5.

D - Correct: Both change when the first node is inserted. See assigned PDF 7, 8, 5.

E - Incorrect: Rear alone changes only in the nonempty case. See assigned PDF 7, 8, 5.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs

if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 8 | lines 35-37

III. Linked list implementation of enqueue and dequeue operations
Enqueue- is inserting a node at the end of a linked list
Dequeue- is deleting the first node in the list

Data Structure and Algorithms module | PDF page 5 | lines 13-24

Step 3: Then add the node at the end of the list. For this we need to check whether the list is empty or not. If list is empty the new node will be the first (Start) node, otherwise navigate forward starting form the first node till last node found and add it next to this node which make the new node the last node.

```
if (start == NULL) // If list is empty, then new node become start node
start = temp;
else {
temp2 = start; // list not empty! Thus temp2 starts from first node
while (temp2->next!= NULL) { // till the current last node found navigates forward
temp2 = temp2->next; // Move to next link in chain
}
temp2->next = temp; // new node (temp) appended at the end of the list
```

Question sources

Mock exam 4 · Question 18

Question 17 | Q00217

Binary search is to operate on an already ordered sequence of comparable keys through an indexed-access interface. Which listed property is required of its search domain, rather than being a particular physical array representation or allocation policy?

A. The array must hold elements of the same data type.

B. All are additional requirements.

C. The array must be contiguous.

D. The search interval must have valid finite bounds.

E. The array must fix its size a prior to population.

F. None is an additional requirement.

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: D. The search interval must have valid finite bounds.

Binary search maintains an interval of candidate indices and repeatedly shrinks it until a match or an empty interval is obtained. That operation requires valid finite bounds. Homogeneous physical storage and contiguity are properties of ordinary array representations, rather than requirements of the abstract interval-halving procedure. Fixing capacity before population is unnecessary: a resizable sequence can be populated first and searched afterward. Hence not all listed properties are requirements.

Why each choice is right or wrong

A - Incorrect: Comparable keys are assumed; identical physical data types are not the abstract domain requirement.

B - Incorrect: All fails because prior fixed allocation is unnecessary.

C - Incorrect: Contiguity is a representation choice rather than the abstract search-domain requirement.

D - Correct: The candidate index domain needs valid finite bounds.

E - Incorrect: A resizable sequence may be populated before searching.

F - Incorrect: Valid finite bounds are a required condition among these choices.

Primary note proof

Study-note section DSA29

> On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Question sources

Mock exam 4 · Question 19

Question 18 | Q00218

What will be the content of the queue (shown from queue front on the left to queue rear on the right), after the given sequence of operations shown below?

```
Queue q = new Queue();
q.enqueue(4);
q.enqueue(8);
if(q.peek() > 5)
q.dequeue();
q.enqueue(1);
q.enqueue(6);
q.dequeue();
```

A. 4, 6

B. 4, 8, 1

C. 8, 1, 6

D. 8, 6

E. 4, 1

Hint

Trace the initial state and each executed operation in order, preserving the requested output layout.

Answer C: 8, 1, 6

After enqueues the queue is [4,8]. Peek examines the front 4, so 4>5 is false and the conditional dequeue does not execute. Enqueues give [4,8,1,6]; the final dequeue removes 4, leaving [8,1,6]. Evidence: assigned PDF pages 7, 6.

Why each choice is right or wrong

A - Incorrect: Drops 8 and 1 without corresponding deletions. See assigned PDF 7, 6.

B - Incorrect: Leaves the removed 4 and omits the enqueued 6. See assigned PDF 7, 6.

C - Correct: Matches the final FIFO sequence. See assigned PDF 7, 6.

D - Incorrect: Incorrectly removes 1 as well as 4. See assigned PDF 7, 6.

E - Incorrect: Drops 8 and 6 and retains the removed 4. See assigned PDF 7, 6.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Mock exam 4 · Question 20

Question 19 | Q00286

What is the tightest listed worst-case time bound for constructing an optimal binary search tree using the standard interval dynamic program that tries every key as root of every interval?

A. O(n)

B. $O(n^3)$

C. O(logn)

D. O (n!)

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: B. $O(n^3)$

This asks about constructing an optimal BST, rather than searching a tree after construction. The standard dynamic program considers quadratically many key intervals. For each interval it tries up to a linear number of candidate roots, combining the relevant subproblem costs. Multiplying those counts gives cubic construction time. The cubic option states that bound; the linear, logarithmic and factorial alternatives do not give the tight construction order.

Why each choice is right or wrong

A - Incorrect: Linear time does not cover all intervals and root candidates.

B - Correct: Quadratic intervals times linear root trials give cubic time.

C - Incorrect: Logarithmic time concerns balanced search paths, not this construction.

D - Incorrect: Factorial is a loose upper bound rather than the tight listed order.

Primary note proof

Study-note section DSA19

> Standard optimal-BST dynamic programming tries each possible root for each key interval: $O(n^2)$ intervals times $O(n)$ roots gives $O(n^3)$ construction.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

> • Binary tree: a tree in which each node has at most two children called left child and right child.
> • Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
> satisfies the following.
> o Every node has a key and no two elements have the same key.
> o The keys in the right subtree are larger than the keys in the root.
> o The keys in the left subtree are smaller than the keys in the root.
> o The left and the right subtrees are also binary search trees.

Question sources

Computer Science model exam · Question 17

University of Gondar model exam · Question 17

Question 20 | Q00288

Which of the following is the best case time complexity of linear search algorithm?

A. O(n'2)

B. O(n)

C. O(1)

D. O(log n)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: O(1)

A match at the very first inspected element ends scanning after one comparison, independent of n. Interpret the options as tight growth orders rather than arbitrary loose Big-O upper bounds. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: Quadratic is not the tight best-case count. See assigned PDF 3.

B - Incorrect: Linear is the complete-scan bound, not the immediate-match cost. See assigned PDF 3.

C - Correct: One first-element comparison is constant. See assigned PDF 3.

D - Incorrect: Logarithmic describes halving, not immediate linear-search success. See assigned PDF 3.

Primary note proof

Study-note section DSA28

> Compare the target with successive elements until equal or exhausted. Unsorted data is allowed; sorted data is allowed too. A first-element match costs Theta(1). An absent target or last-position match needs n comparisons, giving Theta(n) worst time. Uniformly likely successful positions average (n+1)/2 comparisons, still linear.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

> Linear/Sequential searching
> • Works with unsorted List
> • Loop through the list starting at the first element
> • Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Computer Science model exam · Question 19

University of Gondar model exam · Question 19

Question 21 | Q00289

If the elements '1', '2', '3' and '4' are inserted in a queue, what would be order for the removal?

A. 4321

B. 3241

C. 1234

D. None of the above

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: 1234

FIFO removes the four insertions in their original order: 1 then 2 then 3 then 4. Evidence: assigned PDF pages 7.

Why each choice is right or wrong

A - Incorrect: 4321 reverses arrival order. See assigned PDF 7.

B - Incorrect: 3241 puts later arrivals before 1 and 2. See assigned PDF 7.

C - Correct: 1234 preserves insertion order. See assigned PDF 7.

D - Incorrect: C already supplies the FIFO result. See assigned PDF 7.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Computer Science model exam · Question 20

University of Gondar model exam · Question 20

Question 22 | Q00290

What is the disadvantage of array data structure?

A. The amount of memory to be allocated should be known beforehand.

B. Elements of an array can be accessed in constant time.

C. Elements are stored in contiguous memory blocks.

D. Multiple other data structures can be implemented using arrays.

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: The amount of memory to be allocated should be known beforehand.

The module calls array storage static and contrasts it with dynamically allocated nodes. A fixed array requires a capacity in advance; constant access and reuse as a storage implementation are benefits, while contiguous storage alone is not the capacity drawback. Evidence: assigned PDF pages 6, 7, 5.

Why each choice is right or wrong

A - Correct: Preallocated fixed capacity is the disadvantage illustrated by static array implementations. See assigned PDF 6, 7, 5.

B - Incorrect: Constant indexed access is an advantage, not a disadvantage. See assigned PDF 6, 7, 5.

C - Incorrect: Contiguity itself does not state the capacity limitation. See assigned PDF 6, 7, 5.

D - Incorrect: Supporting other structures is useful rather than a disadvantage. See assigned PDF 6, 7, 5.

Primary note proof

Study-note section DSA02

> A fixed array has a chosen capacity; the choice need not occur at compile time. A resizable array allocates a larger block and copies when needed. Inserting in the middle shifts later elements, so it costs linear time. Appending into spare capacity is constant; a resizing append can be linear, though repeated geometric growth gives amortized constant append cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 24-42

> I. Array implementation (static memory).
> Push(n): Adding item at top

```
top is the current top of stack and n is its
maximum size
Begin
if (top == n) then stack is full;
else
top := top+1;
stack[top] := item;
end:
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;
```

Data Structure and Algorithms module | PDF page 7 | lines 18-23

I. Simple Array implementation (static memory).
A simple array structure can be used to implement enqueue & dequeue operations. Thus item will
be added at the end of the array whereas dequeue operation removes items from the front. In simple
array implementation, to add item it checks whether there is room at the rear or not. Therefore, even though there are free released spaces at the front ot the array the queue implementation might
generate overflow message. This is the key drawback to use simple array to implement Queue.

Data Structure and Algorithms module | PDF page 5 | lines 3-8

While adding a node to a list, the chain needs to be rearranged properly so that the each nodes are connected to each other. It is possible to add a node anywhere in the chain (at the start, middle or end). Let's consider adding a node at the end of the list:
Step 1: First we need to reserve a space for the new node using the keyword new.
Node *temp = new *Node; // Reserving a space for temporary pointer which points to new node;

Question sources

Computer Science model exam · Question 21

University of Gondar model exam · Question 21

Question 23 | Q00291

For selection sort on n elements, n>=2, count only exchanges between distinct positions and omit the final one-element suffix. What is the maximum number of swaps?

A. n - 1

B. 1

C. n

D. n - 2

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. n - 1

Selection sort places the minimum remaining element into each successive position. After n-1 such placements, the last position is determined automatically. Each placement needs at most one exchange between distinct positions, so the maximum is n-1. This count excludes self-swaps and the redundant final-position call that some implementations include. The choice of input can require an exchange on every one of those meaningful passes.

Why each choice is right or wrong

A - Correct: There are n-1 meaningful placements, each requiring at most one exchange.

B - Incorrect: More than one exchange can be needed for larger n.

C - Incorrect: n counts an extra redundant final swap call under another convention.

D - Incorrect: n-2 omits a possible exchange on a meaningful pass.

Primary note proof

Study-note section DSA33

> At position i, scan the remaining suffix for its minimum, then exchange it with A[i]. There are n-1 meaningful placements and n(n-1)/2 comparisons regardless of initial order. At most n-1 exchanges of distinct positions are needed. A loop that also swaps the last item with itself can count n swap calls.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 44-48

> B. Selection Sort
> Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
> • How many comparisons? => O(n2
>) and How many swaps? => O(n)

Question sources

Computer Science model exam · Question 22

University of Gondar model exam · Question 22

Question 24 | Q00293

In an infix-to-prefix conversion algorithm that temporarily holds pending operators and parentheses using last-in, first-out access, which abstract data type supplies that access discipline?

A. Stack

B. Queue

C. Binary tree

D. Linked list

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. Stack

Pending operators must be retrieved in reverse order when grouping or precedence requires them to be emitted. A stack provides that last-in, first-out discipline. An expression tree is another possible representation for conversion, but it is not the requested pending-operator ADT in this specified algorithm. A queue retrieves the oldest entry first, while a linked list alone describes storage links without prescribing the needed access policy.

Why each choice is right or wrong

A - Correct: A stack retrieves the latest pending operator first.

B - Incorrect: A queue uses first-in, first-out access.

C - Incorrect: An expression tree can support another conversion method but is not this pending-operator ADT.

D - Incorrect: A linked list can implement storage but alone does not specify LIFO behavior.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA13

To evaluate postfix, push operands. At a binary operator, pop right operand first, then left, and push left op right. Build an expression tree for either prefix (preorder) or postfix (postorder); a stack is useful but is not the only possible conversion representation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 46-52

Applications of Stack
• It is very useful to evaluate arithmetic expressions. (Postfix Expressions)
• Infix to Postfix Transformation
• It is useful during the execution of recursive programs
• A Stack is useful for designing the compiler in operating system to store local variables inside a function block.
• A stack (memory stack) can be used in function calls including recursion.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

• Reversing Data and Reverse a List
• Convert Decimal to Binary
• Parsing - It is a logic that breaks into independent pieces for further processing
• Backtracking
Examples:
1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Question sources

Computer Science model exam · Question 24

University of Gondar model exam · Question 24

Question 25 | Q00294

Which one of the following is the process of inserting an element in the stack?

A. Insert

B. Add

C. Push

D. None of the above

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Push

The stack terminology explicitly names adding an element Push. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Incorrect: Insert is a generic action rather than the given operation name. See assigned PDF 6.

B - Incorrect: Add is descriptive rather than the module stack name. See assigned PDF 6.

C - Correct: Push is defined as adding an element to the stack. See assigned PDF 6.

D - Incorrect: C is explicitly defined, so none is false. See assigned PDF 6.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science model exam · Question 25

University of Gondar model exam · Question 25

Question 26 | Q00371

How many cases are there, which are used to compare various data structure's execution time in a relative manner?

A. 2

B. 3

C. 4

D. 5

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: Three: best, average, worst.

DSA05 names best, average, and worst as the ordinary cases for comparing costs on inputs of equal size. Average additionally needs an input distribution. Counting these cases gives three. This is a classification of input cases, rather than a count of resources, data structures, or asymptotic bound notations.

Why each choice is right or wrong

A - Incorrect: Two omits one of the three named categories: best, average, and worst case.

B - Correct: The note lists three ordinary fixed-size input cases: best, average, and worst.

C - Incorrect: Four does not match the three ordinary input-case categories listed in DSA05.

D - Incorrect: Five adds categories outside the stated best, average, and worst input-case classification.

Primary note proof

Study-note section DSA-S01

> Fix the input size before comparing cases. Best-case cost is the minimum resource use over legal inputs of that size; worst-case cost is the maximum. Average-case cost averages costs using a stated input distribution. These three input-case functions can each be described using upper, lower or tight asymptotic bounds; the case name does not select the bound notation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 33-37

> Algorithm analysis refers to the process of determining the amount of computing time and storage
> space required by different algorithms. It used to produce a function T (n) that describes the algorithm in terms of the operations performed in order to measure the complexityof the algorithm.
> Complexity Analysis is the systematic study of the cost of computation, measured either in time units or in operations performed, or in the amount of storage space required.

Question sources

Data Structures & Algorithms practice questions · Question 1

Question 27 | Q00372

Which case of data structure operation takes maximum time?

A. Worst Case

B. Average Case

C. Best Case

D. None of the above

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer A: Worst case

DSA05 distinguishes best, average, and worst over inputs of the same size. Worst selects the largest runtime; best selects the smallest, while average takes an expectation under a distribution. Therefore Worst Case directly matches maximum time. This input-case distinction is independent of whether the resulting cost function is described with O, Ω, or Θ notation.

Why each choice is right or wrong

A - Correct: Worst case takes the maximum operation cost over valid inputs of the same size.

B - Incorrect: Average case computes an expected cost under a distribution instead of selecting the maximum.

C - Incorrect: Best case selects the minimum cost, the opposite extremum from the question.

D - Incorrect: Worst Case is offered and directly identifies the requested maximum-time input case.

Primary note proof

Study-note section DSA-S01

> Fix the input size before comparing cases. Best-case cost is the minimum resource use over legal inputs of that size; worst-case cost is the maximum. Average-case cost averages costs using a stated input distribution. These three input-case functions can each be described using upper, lower or tight asymptotic bounds; the case name does not select the bound notation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 33-37

> Algorithm analysis refers to the process of determining the amount of computing time and storage
> space required by different algorithms. It used to produce a function T (n) that describes the algorithm in terms of the operations performed in order to measure the complexityof the algorithm.
> Complexity Analysis is the systematic study of the cost of computation, measured either in time units or in operations performed, or in the amount of storage space required.

Question sources

Data Structures & Algorithms practice questions · Question 2

Question 28 | Q00375

Which listed characteristic is a binary-tree shape condition rather than a general functional-correctness or resource-cost criterion for arbitrary data-structure operations? Interpret completeness as the complete-tree shape property.

A. Completeness

B. Correctness

C. Time Complexity

D. Space Complexity

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. Completeness

Correctness asks whether operations produce their specified results. Time and space complexity measure operation counts and memory use under a resource model. Completeness, under the explicit meaning supplied here, instead describes how a binary tree's levels are filled: earlier levels are full and the last fills from the left. It is a particular shape condition, rather than a universal assessment criterion for every data structure.

Why each choice is right or wrong

A - Correct: Completeness describes a particular binary-tree shape.

B - Incorrect: Correctness assesses whether an operation meets its specification.

C - Incorrect: Time complexity assesses operation cost.

D - Incorrect: Space complexity assesses memory cost.

Primary note proof

Study-note section DSA18

> Binary means at most two children. Full means each node has zero or two children. Complete means all earlier levels are filled and the last fills from left. Perfect means all leaves share a level and every internal node has two children. Balanced means height stays logarithmic under a stated rule; it does not require a perfect shape.

Study-note section DSA04

> Finiteness means termination after finitely many steps for every valid input. Definiteness means each instruction has one interpretation; effectiveness means each step is executable. Correctness means meeting the stated result for every valid input. A finite program can still return the wrong answer.

Study-note section DSA05

> Time counts chosen operations; space counts memory, including live recursive frames.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 33-37

> Algorithm analysis refers to the process of determining the amount of computing time and storage
> space required by different algorithms. It used to produce a function T (n) that describes the algorithm in terms of the operations performed in order to measure the complexityof the algorithm.
> Complexity Analysis is the systematic study of the cost of computation, measured either in time units or in operations performed, or in the amount of storage space required.

Question sources

Data Structures & Algorithms practice questions · Question 5

Question 29 | Q00377

Which abstract data types permit deleting an element at the front and inserting an element at the rear? Include types that also permit additional endpoint operations. Select all that apply.

A. Stacks

B. Queues

C. Deques

D. Binary search tree

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B, C: B. Queues; C. Deques

An ordinary queue supplies exactly the stated FIFO operations: insertion at the rear and removal at the front. A deque allows insertion and removal at both ends, which includes those same two operations as a subset. The question explicitly includes types with additional endpoint capabilities, so both qualify. A stack exposes a single LIFO top, and a BST organizes access by keys rather than front and rear.

Why each choice is right or wrong

A - Incorrect: A stack exposes a single LIFO top rather than these FIFO endpoints.

B - Correct: A queue enqueues at rear and dequeues at front.

C - Correct: A deque includes both requested operations among its endpoint capabilities.

D - Incorrect: A BST uses key-based tree operations rather than front/rear endpoints.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Study-note section DSA16

> A deque permits both-end insertion/removal without guaranteeing middle access. Endpoint-linked operations are constant but middle indexing is linear. A circular-array deque may support indexed access. Dequeue names removal; deque names the structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 8 | lines 38-40

> Deque (pronounced as Deck): is a Double Ended Queue in which insertion and deletion can occur at either end.
> Priority Queue:- is a queue where each data has an associated key provided at insertion time.

Question sources

Data Structures & Algorithms practice questions · Question 7

Question 30 | Q00379

Which algorithms have tight worst-case time Theta(n^2)? Use ordinary array comparison-sort implementations; radix sort uses counting-based passes with fixed digit count and fixed radix. Select all that apply.

A. Merge sort

B. Insertion sort

C. Quick sort

D. Radix sort

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer B, C: B. Insertion sort; C. Quick sort

Insertion sort on reverse input shifts a growing sorted prefix, totaling quadratic work. Quicksort repeatedly choosing an extreme pivot also accumulates quadratic partition work. Ordinary merge sort instead performs linear work at logarithmically many levels. Under the explicitly fixed digit count and radix, radix sorting is linear in n. Tight bounds distinguish these actual growth rates from the looser quadratic upper bounds that faster algorithms also satisfy.

Why each choice is right or wrong

A - Incorrect: Ordinary merge sort has tight n log n worst time.

B - Correct: Reverse-order insertion gives quadratic shifts.

C - Correct: Repeated extreme quicksort pivots give quadratic work.

D - Incorrect: Fixed-width, fixed-radix counting passes give linear time.

Primary note proof

Study-note section DSA34

> Reverse order causes 1+2+...+(n-1) shifts: quadratic time.

Study-note section DSA36

> Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work.

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA39

> With d digits and radix r, counting-based passes take O(d(n+r)) time and O(n+r) auxiliary storage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 34-43

> A. Insertion Sort
> The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
> them in order. Then add 3rd
> item to the relatively sorted portion and find its correct location
> (relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
> in the list.
> • How many comparisons? => O(n2
>) and How many swaps? => O(n2
>)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Data Structures & Algorithms practice questions · Question 9

Question 31 | Q00380

In which sorting algorithm does the largest element "bubble" to the end of the list at each pass?

A. Bubble sort

B. Selection sort

C. Insertion sort

D. Heap sort

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Bubble sort

Repeated adjacent out-of-order swaps push the largest item toward the final position in a left-to-right pass. This is exactly the bubble-sort mechanism. Evidence: assigned PDF pages 4, 3, 11.

Why each choice is right or wrong

A - Correct: Adjacent comparisons bubble the largest to its proper end. See assigned PDF 4, 3, 3, 11.

B - Incorrect: Selection finds a minimum and swaps it into position rather than bubbling through adjacent pairs. See assigned PDF 4, 3, 3, 11.

C - Incorrect: Insertion inserts each new element into a sorted prefix. See assigned PDF 4, 3, 3, 11.

D - Incorrect: Heap sort removes a root maximum using tree adjustment. See assigned PDF 4, 3, 3, 11.

Primary note proof

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right. The largest remaining value moves to the final unsorted position after one left-to-right pass. Shrink the next pass by one position. Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Study-note section DSA33

> At position i, scan the remaining suffix for its minimum, then exchange it with A[i].

Study-note section DSA34

> For each remaining item, save its value, shift larger preceding values right, and insert it into the gap.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

> C. Bubble Sort
> The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
> the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
> • How many comparisons? => O(n2
>) and How many swaps? => O(n2
>)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

> B. Selection Sort
> Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
> • How many comparisons? => O(n2
>) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

> A. Insertion Sort
> The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
> them in order. Then add 3rd

item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Question sources

Data Structures & Algorithms practice questions · Question 10

Question 32 | Q00381

Which algorithms have tight best-case time Theta(n)? Use early-exit bubble sort, ordinary quicksort, ordinary array mergesort without merge-skipping, and standard insertion sort. Select all that apply.

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer A, D: A. Bubble sort; D. Insertion sort

Early-exit bubble sort terminates after one complete no-swap pass on already sorted input, so its best case is linear. Insertion sort also checks each new item once without shifting it on sorted input. Ordinary quicksort still partitions across logarithmically many balanced levels in its best case, and ordinary mergesort still performs its merges. Thus exactly the bubble and insertion implementations have the requested linear best case.

Why each choice is right or wrong

A - Correct: The no-swap early exit gives a linear sorted-input pass.

B - Incorrect: Ordinary quicksort has n log n best growth.

C - Incorrect: Without merge-skipping, mergesort still uses n log n work.

D - Correct: Sorted input needs one ordering check per insertion pass.

Primary note proof

Study-note section DSA32

> Early stopping makes a sorted input linear: one no-swap pass finishes.

Study-note section DSA34

> On sorted input each pass needs one ordering check and no shifts: linear time.

Study-note section DSA36

> Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work.

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Related reading (context, not primary note proof)

Study-note section DSA33

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

> C. Bubble Sort
> The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
> the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
> • How many comparisons? => O(n2
>) and How many swaps? => O(n2
>)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Variant 1 | Alternative version 1

Which algorithms have tight best-case time Theta(n)? Use early-exit bubble sort, standard selection sort, ordinary array mergesort without merge-skipping, and standard insertion sort. Select all that apply.

A. Bubble sort

B. Selection sort

C. Merge sort

D. Insertion sort

Answer A, D: A. Bubble sort; D. Insertion sort

Early-exit bubble sort needs one no-swap pass on sorted input, while insertion sort performs one successful ordering check per new item with no shifts. Both have linear best time. Selection sort still scans every remaining suffix and makes quadratic comparisons, even on sorted input. Ordinary mergesort retains logarithmically many levels of merging. Consequently A and D are the complete selection for this variant.

Why each choice is right or wrong

A - Correct: An early-exit no-swap pass makes optimized bubble sort linear on sorted input.

B - Incorrect: Selection sort still scans each suffix and makes quadratic comparisons even when sorted.

C - Incorrect: Ordinary merge sorting does n log n work across input orders.

D - Correct: Insertion sort is linear on sorted input because each pass needs no shifts.

Primary note proof

Study-note section DSA32

> Early stopping makes a sorted input linear: one no-swap pass finishes.

Study-note section DSA34

> On sorted input each pass needs one ordering check and no shifts: linear time.

Study-note section DSA33

> There are n-1 meaningful placements and n(n-1)/2 comparisons regardless of initial order.

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Related reading (context, not primary note proof)

Study-note section DSA36

Question sources

Data Structures & Algorithms practice questions · Question 11

Computer Science MCQ practice collection · Question 454

Question 33 | Q00382

Which ordinary iterative array sorting implementations use constant auxiliary element storage? Use buffered mergesort for C. Select all that apply.

A. Bubble sort

B. Selection sort

C. Merge sort

D. Insertion sort

Hint

Apply the naming, type and syntax rules before deciding whether an option can be used as stated.

Answer A, B, D: A. Bubble sort; B. Selection sort; D. Insertion sort

Bubble sort reuses one temporary element for each adjacent swap. Selection sort retains the minimum position while scanning, then exchanges elements using the same fixed-size temporary storage. Insertion sort saves one item and shifts preceding entries within the original array before filling the gap. The indices and saved elements stay fixed in number as n grows, giving constant auxiliary storage for A, B and D. Buffered mergesort instead needs a linear auxiliary buffer.

Why each choice is right or wrong

A - Correct: Adjacent swaps reuse a single temporary element and a fixed number of indices.

B - Correct: The suffix scan retains a minimum index; the exchange needs only fixed-size temporary storage.

C - Incorrect: The specified buffered array mergesort allocates a linear auxiliary merge buffer.

D - Correct: One saved item and fixed-size indices suffice while larger entries shift within the original array.

Primary note proof

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right.

Study-note section DSA34

> For each remaining item, save its value, shift larger preceding values right, and insert it into the gap.

Study-note section DSA33

> At position i, scan the remaining suffix for its minimum, then exchange it with A[i].

Study-note section DSA37

> Ordinary array merging needs a linear auxiliary buffer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Data Structures & Algorithms practice questions · Question 12

Question 34 | Q00383

Which sorting algorithm uses a pivot element to partition the list into two sublists?

A. Bubble sort

B. Selection sort

C. Quick sort

D. Insertion sort

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Quick sort

The source quicksort procedure chooses a pivot and partitions around it before recursively sorting both sides. Evidence: assigned PDF pages 11, 4, 3.

Why each choice is right or wrong

A - Incorrect: Bubble uses adjacent exchanges without a pivot partition. See assigned PDF 11, 4, 3, 3.

B - Incorrect: Selection chooses the minimum for its next position. See assigned PDF 11, 4, 3, 3.

C - Correct: Quicksort explicitly chooses and positions a pivot. See assigned PDF 11, 4, 3, 3.

D - Incorrect: Insertion extends a sorted prefix rather than partitioning by a pivot. See assigned PDF 11, 4, 3, 3.

Primary note proof

Study-note section DSA36

> Choose a pivot, partition, then recursively sort the pieces. Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work. Expected n log n needs an input/random-pivot model. Random and median-of-three choices have different tradeoffs; neither is universally best.

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right.

Study-note section DSA34

> Begin with the first element already sorted.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

> ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).

Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Data Structures & Algorithms practice questions · Question 13

Question 35 | Q00384

Which sorting implementations preserve the relative input order of equal keys? Use bubble sort with strict adjacent swaps, standard exchange-based selection sort, ordinary in-place quicksort, and mergesort with left-first tie handling. Select all that apply.

A. Bubble sort

B. Selection sort

C. Quick sort

D. Merge sort

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A, D: A. Bubble sort; D. Merge sort

Strict bubble swaps never exchange equal adjacent keys, so equal records retain their input order. Left-first handling during merging likewise preserves the order of ties from the left and right runs. Standard selection sort can reverse ties through a distant minimum exchange, and ordinary quicksort partitioning does not guarantee stable order. The specified implementation details therefore make bubble and merge the two stable choices.

Why each choice is right or wrong

A - Correct: Strict adjacent swaps preserve equal-key order.

B - Incorrect: A distant minimum exchange can reverse equal records.

C - Incorrect: Ordinary in-place partitioning does not guarantee stability.

D - Correct: Taking the left run first on ties preserves equal-key order.

Primary note proof

Study-note section DSA32

> Strict swaps preserve equal-key order.

Study-note section DSA37

> With left-first tie handling, equal keys retain order.

Study-note section DSA33

> Selection suits expensive record moves with cheap key comparisons, not every large dataset. It is unstable: exchanging first 2a with 1 in [2a,2b,1] gives [1,2b,2a]. Constant extra element storage is shared by several sorts.

Study-note section DSA31

> stack usually log n average, n worst; unstable

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in
the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Data Structures & Algorithms practice questions · Question 14

Question 36 | Q00385

Which algorithm has tight Theta(n log n) time in both the average and worst cases? Use ordinary array implementations; radix sort uses fixed digit count and fixed radix.

A. Bubble sort

B. Selection sort

C. Merge sort

D. Radix sort

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: C. Merge sort

Ordinary mergesort performs linear merging at each of logarithmically many levels, giving n log n work regardless of input order. Bubble and selection have quadratic average and worst time. Fixed-width, fixed-radix counting passes instead give radix sort linear time, so radix does not have the requested tight n log n growth. Specifying tight bounds excludes faster methods that would also satisfy the looser Big-O upper bound.

Why each choice is right or wrong

A - Incorrect: Bubble has quadratic average and worst time.

B - Incorrect: Selection makes quadratic comparisons in every order.

C - Correct: Ordinary mergesort has tight n log n average and worst time.

D - Incorrect: With fixed digits and radix, radix sorting has tight linear time.

Primary note proof

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA39

> With d digits and radix r, counting-based passes take O(d(n+r)) time and O(n+r) auxiliary storage.

Study-note section DSA32

> Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Study-note section DSA33

> There are n-1 meaningful placements and n(n-1)/2 comparisons regardless of initial order.

Related reading (context, not primary note proof)

Study-note section DSA34

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Variant 1 | Alternative version 1

Under ordinary array comparison-sort implementations, which algorithm has tight Theta(n log n) time in every input case? Exclude optimizations that skip work on already ordered input.

A. Bubble sort

B. Selection sort

C. Merge sort

D. Insertion sort

Answer C: C. Merge sort

Buffered mergesort has linear merge work at logarithmically many levels in every input case, giving tight n log n average and worst growth. Bubble, selection and insertion have quadratic worst time, and their usual average time is also quadratic. Quicksort's extreme partitions can violate the requested worst-case bound, whereas buffered mergesort meets it.

Why each choice is right or wrong

A - Incorrect: Bubble has quadratic average and worst time.

B - Incorrect: Selection always makes quadratic comparisons.

C - Correct: Mergesort has the required tight average and worst bounds.

D - Incorrect: Insertion has quadratic average and worst time.

Primary note proof

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA32

> Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Study-note section DSA33

> There are n-1 meaningful placements and n(n-1)/2 comparisons regardless of initial order.

Study-note section DSA34

> Reverse order causes 1+2+...+(n-1) shifts: quadratic time.

Related reading (context, not primary note proof)

Study-note section DSA39

Question sources

Data Structures & Algorithms practice questions · Question 15

Computer Science MCQ practice collection · Question 438

Question 37 | Q00386

For a large array of nonnegative integer keys with fixed digit count and fixed radix, which listed ordinary algorithm can achieve linear time using counting-based digit passes and O(n+r) auxiliary storage?

A. Bubble sort

B. Selection sort

C. Radix sort

D. Insertion sort

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: C. Radix sort

With digit count d and radix r fixed, the radix cost O(d(n+r)) reduces to linear growth in n. Its counting-based stable passes use auxiliary storage for output and digit counts. The key assumptions make this an eligible radix-sorting workload rather than an unspecified large-list claim. Bubble, selection and insertion use constant extra element storage in their ordinary forms and can require quadratic work.

Why each choice is right or wrong

A - Incorrect: Bubble uses adjacent comparisons and can require quadratic work.

B - Incorrect: Selection uses quadratic comparisons and constant extra element storage.

C - Correct: Radix uses the stated digit passes and auxiliary storage.

D - Incorrect: Insertion uses prefix shifts and can require quadratic work.

Primary note proof

Study-note section DSA39

> With d digits and radix r, counting-based passes take O(d(n+r)) time and O(n+r) auxiliary storage.

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right.

Study-note section DSA34

> For each remaining item, save its value, shift larger preceding values right, and insert it into the gap.

Study-note section DSA33

> At position i, scan the remaining suffix for its minimum, then exchange it with A[i].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Data Structures & Algorithms practice questions · Question 16

Question 38 | Q00387

Which listed sorting algorithm orders fixed-length strings by stably processing individual character positions, without using whole-string comparison operations? Assume an ordered finite character alphabet and LSD passes.

A. Bubble sort

B. Selection sort

C. Merge sort

D. Radix sort

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: D. Radix sort

Strings can be compared lexicographically, but the stated question asks for a digit-based mechanism rather than whole-key comparisons. LSD radix sorting treats each character position as a digit, stably sorting from the last position toward the first. Stability retains the ordering established by earlier passes. Bubble, selection and merge use comparisons between keys in their ordinary forms, so they do not implement this specified character-pass mechanism.

Why each choice is right or wrong

A - Incorrect: Ordinary bubble sort compares adjacent whole keys.

B - Incorrect: Ordinary selection chooses a minimum through key comparisons.

C - Incorrect: Ordinary merge compares the fronts of sorted key runs.

D - Correct: LSD radix uses stable character-position passes from least significant to most significant.

Primary note proof

Study-note section DSA39

> LSD radix sorting handles fixed-width digit keys by stably sorting least-significant digit first, then moving left. Stable intermediate passes preserve the ordering already established by less-significant digits. With d digits and radix r, counting-based passes take O(d(n+r)) time and O(n+r) auxiliary storage.

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right.

Study-note section DSA33

> At position i, scan the remaining suffix for its minimum, then exchange it with A[i].

Study-note section DSA37

> Split into halves, sort them, then merge by advancing the smaller current front.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

> C. Bubble Sort
> The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
> the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
> • How many comparisons? => O(n2

) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Data Structures & Algorithms practice questions · Question 17

Question 39 | Q00388

How many passes does an insertion sort algorithm consist of?

A. N

B. N-1

C. N+1

D. N2

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: N-1

The first element is already a sorted prefix. Each of the remaining N-1 elements is inserted once: the source starts with the first two, then adds the third, and so on. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: N counts the initial singleton as an insertion pass. See assigned PDF 3.

B - Correct: There is one insertion for each position 2 through N. See assigned PDF 3.

C - Incorrect: N+1 adds two passes with no corresponding elements. See assigned PDF 3.

D - Incorrect: N squared is a work bound, not the count of outer passes. See assigned PDF 3.

Primary note proof

Study-note section DSA34

> Begin with the first element already sorted. For each remaining item, save its value, shift larger preceding values right, and insert it into the gap. That gives n-1 insertion passes. On sorted input each pass needs one ordering check and no shifts: linear time. Reverse order causes 1+2+...+(n-1) shifts: quadratic time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 34-43

> A. Insertion Sort
> The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
> them in order. Then add 3rd
> item to the relatively sorted portion and find its correct location
> (relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
> in the list.
> • How many comparisons? => O(n2
>) and How many swaps? => O(n2
>)

Question sources

Data Structures & Algorithms practice questions · Question 18

Question 40 | Q00389

What is the running time of an insertion sort algorithm if the input is pre-sorted?

A. O(N2)

B. O(N log N)

C. O(N)

D. O(M log N)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: O(N)

For sorted input every new item already belongs after its predecessor. Each of N-1 prefix extensions needs constant comparison work and no shifts, giving Theta(N). Big-O choices are interpreted as tight orders. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: Quadratic is the adverse-input bound, not the tight sorted-input cost. See assigned PDF 3.

B - Incorrect: N log N adds work not performed by this insertion process. See assigned PDF 3.

C - Correct: One constant check per extension gives linear work. See assigned PDF 3.

D - Incorrect: M is undefined and no M-dependent process is given. See assigned PDF 3.

Primary note proof

Study-note section DSA34

> Begin with the first element already sorted. For each remaining item, save its value, shift larger preceding values right, and insert it into the gap. That gives n-1 insertion passes. On sorted input each pass needs one ordering check and no shifts: linear time. Reverse order causes 1+2+...+(n-1) shifts: quadratic time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 34-43

> A. Insertion Sort
> The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
> them in order. Then add 3rd
> item to the relatively sorted portion and find its correct location
> (relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
> in the list.
> • How many comparisons? => O(n2
>) and How many swaps? => O(n2
>)

Question sources

Data Structures & Algorithms practice questions · Question 19

Question 41 | Q00390

Which of the following examples represent the worst case input for an insertion sort?

A. array in sorted order

B. array sorted in reverse order

C. normal unsorted array

D. large array

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: array sorted in reverse order

Reverse order forces each inserted item to move through the whole preceding prefix. The movement total 1+2+...+(n-1)=n(n-1)/2 is quadratic. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: Sorted input needs no shifts. See assigned PDF 3.

B - Correct: Each next element is smaller than every element in the prefix. See assigned PDF 3.

C - Incorrect: An arbitrary unsorted array need not maximize all insertion distances. See assigned PDF 3.

D - Incorrect: Large size alone does not determine relative ordering or maximal shifts. See assigned PDF 3.

Primary note proof

Study-note section DSA34

> Begin with the first element already sorted. For each remaining item, save its value, shift larger preceding values right, and insert it into the gap. That gives n-1 insertion passes. On sorted input each pass needs one ordering check and no shifts: linear time. Reverse order causes 1+2+...+(n-1) shifts: quadratic time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 34-43

> A. Insertion Sort
> The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
> them in order. Then add 3rd
> item to the relatively sorted portion and find its correct location
> (relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item

in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Data Structures & Algorithms practice questions · Question 20

Question 42 | Q00391

Which scenario specifically favors selection sort's small number of record exchanges, when records themselves are moved and comparisons are cheap? Interpret large values as large records and small keys as inexpensive comparison keys.

A. The input is already sorted

B. A large file has to be sorted

C. Large values need to be sorted with small keys

D. Small values need to be sorted with large keys

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: C. Large values need to be sorted with small keys

Selection sort scans suffixes and makes quadratic comparisons regardless of the initial order, but it performs at most one meaningful exchange per placement. That tradeoff can be useful when physically moving a large record is expensive while comparing its small key is cheap. A large file or sorted input alone does not offer this advantage. Expensive large-key comparisons instead make selection's many comparisons less attractive.

Why each choice is right or wrong

A - Incorrect: Sorted order does not reduce selection's quadratic comparison count.

B - Incorrect: Large file size alone does not identify the expensive-move advantage.

C - Correct: Large records with cheap keys match the many-comparisons/few-exchanges tradeoff.

D - Incorrect: Expensive comparisons with cheap moves do not favor that tradeoff.

Primary note proof

Study-note section DSA33

At position i, scan the remaining suffix for its minimum, then exchange it with A[i]. There are n-1 meaningful placements and n(n-1)/2 comparisons regardless of initial order. At most n-1 exchanges of distinct positions are needed. A loop that also swaps the last item with itself can count n swap calls.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Data Structures & Algorithms practice questions · Question 21

Question 43 | Q00392

What is the advantage of selection sort over other sorting techniques?

A. It requires no additional storage space

B. It is scalable

C. It works best for inputs which are already sorted

D. It is faster than any other sorting technique

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Constant extra element storage, conventionally called in-place sorting.

DSA33 says selection uses constant extra element storage, which is the conventional meaning of A's no-additional-storage phrasing. It still needs a few temporary values, so literally zero bytes would be inaccurate. Its quadratic comparisons do not improve on already sorted input, and the note gives no universal scalability or fastest-algorithm claim. A names the valid conventional benefit.

Why each choice is right or wrong

A - Correct: Interpreted conventionally as in-place, selection uses constant extra element storage rather than an auxiliary array.

B - Incorrect: Quadratic comparisons do not provide an unqualified scalability advantage for growing datasets.

C - Incorrect: Selection makes the same quadratic comparisons even on already sorted input.

D - Incorrect: No universal fastest claim follows; selection trades few moves for many comparisons.

Primary note proof

Study-note section DSA33

> Selection suits expensive record moves with cheap key comparisons, not every large dataset. It is unstable: exchanging first 2a with 1 in [2a,2b,1] gives [1,2b,2a]. Constant extra element storage is shared by several sorts.

Study-note section DSA33

> There are n-1 meaningful placements and n(n-1)/2 comparisons regardless of initial order.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 44-48

> B. Selection Sort
> Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
> • How many comparisons? => O(n2
>) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

> A. Insertion Sort
> The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
> them in order. Then add 3rd
> item to the relatively sorted portion and find its correct location
> (relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item

in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Data Structures & Algorithms practice questions · Question 22

Question 44 | Q00393

What is an external sorting algorithm?

A. Algorithm that uses tape or disk during the sort

B. Algorithm that uses main memory during the sort

C. Algorithm that involves swapping

D. Algorithm that are considered 'in place'

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Sorting using external storage such as disk or tape.

DSA37 defines external sorting through datasets beyond main memory and buffered disk runs. Option A captures this distinguishing external-storage use. Main memory may still hold buffers, but using it alone does not define external sorting. Swapping and being in place are separate implementation properties and do not identify whether the working dataset resides externally.

Why each choice is right or wrong

A - Correct: Tape or disk stores external runs when the working dataset does not fit wholly in main memory.

B - Incorrect: Main-memory use alone does not distinguish external sorting; external algorithms also use memory for buffers.

C - Incorrect: Swapping is an operation used by many sorts and does not define external storage use.

D - Incorrect: In-place concerns auxiliary element storage, independently of whether sorting accesses an external dataset.

Primary note proof

Study-note section DSA37

> Top-down mergesort recurses; bottom-up merges runs of widths 1,2,4,... iteratively. In-place merging variants exist with different tradeoffs; linear merge names an operation, not a universally standardized sort variant. Internal sorting fits the working dataset in memory. External sorting builds sorted runs and merges buffered streams from disk, aiming to reduce I/O. Reverse order alone does not characterize maximum merge comparisons.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 33-37

> Algorithm analysis refers to the process of determining the amount of computing time and storage
> space required by different algorithms. It used to produce a function T (n) that describes the algorithm in terms of the operations performed in order to measure the complexityof the algorithm.
> Complexity Analysis is the systematic study of the cost of computation, measured either in time units or in operations performed, or in the amount of storage space required.

Question sources

Data Structures & Algorithms practice questions · Question 23

Question 45 | Q00395

What is the worst case complexity of bubble sort?

A. O(nlogn)

B. O(logn)

C. O(n)

D. O(n2)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer D: O(n2)

An adversarial input requires repeated passes over adjacent pairs; the total comparisons and swaps are quadratic as stated in the module. Evidence: assigned PDF pages 4.

Why each choice is right or wrong

A - Incorrect: n log n understates the tight adverse-input work. See assigned PDF 4.

B - Incorrect: Logarithmic growth cannot account for repeated full passes. See assigned PDF 4.

C - Incorrect: Linear describes at most a single pass. See assigned PDF 4.

D - Correct: The source explicitly gives O(n2) comparisons and swaps. See assigned PDF 4.

Primary note proof

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right. The largest remaining value moves to the final unsorted position after one left-to-right pass. Shrink the next pass by one position. Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

> C. Bubble Sort
> The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
> the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
> • How many comparisons? => O(n2
>) and How many swaps? => O(n2
>)

Question sources

Data Structures & Algorithms practice questions · Question 25

Question 46 | Q00396

What is the average case complexity of bubble sort?

A. O(nlogn)

B. O(logn)

C. O(n)

D. O(n2)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer D: O(n^2) under the usual average-case model.

DSA32 identifies quadratic average and worst work from repeated adjacent comparisons. Under the usual random-permutation average model, bubble sort therefore takes $\Theta(n^2)$, which implies $O(n^2)$. Early stopping makes an already sorted input linear, but that is the optimized best case rather than the ordinary average over input arrangements.

Why each choice is right or wrong

A - Incorrect: Quadratic average work grows faster than n log n, so this bound is too small.

B - Incorrect: Repeated adjacent passes do not have logarithmic average growth under the usual input model.

C - Incorrect: Linear work is the optimized sorted-input best case, not ordinary average-case bubble sorting.

D - Correct: Quadratic average work is bounded by $O(n^2)$, matching the adjacent-pass analysis in DSA32.

Primary note proof

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right. The largest remaining value moves to the final unsorted position after one left-to-right pass. Shrink the next pass by one position. Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Study-note section DSA-S01

> Fix the input size before comparing cases. Best-case cost is the minimum resource use over legal inputs of that size; worst-case cost is the maximum. Average-case cost averages costs using a stated input distribution. These three input-case functions can each be described using upper, lower or tight asymptotic bounds; the case name does not select the bound notation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Data Structures & Algorithms practice questions · Question 26

Question 47 | Q00397

What is the auxiliary space complexity of merge sort?

A. O(1)

B. O(log n)

C. O(n)

D. O(n log n)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: O(n)

Merging both halves needs an auxiliary array holding up to n elements. Recursion contributes O(log n) frames, dominated by O(n) temporary element storage in the ordinary implementation. Evidence: assigned PDF pages 11.

Why each choice is right or wrong

A - Incorrect: Constant space contradicts the full merge array. See assigned PDF 11.

B - Incorrect: Logarithmic describes recursion depth alone. See assigned PDF 11.

C - Correct: A merge buffer proportional to n dominates auxiliary storage. See assigned PDF 11.

D - Incorrect: n log n counts cumulative work across levels, not simultaneously live storage when buffers are reused. See assigned PDF 11.

Primary note proof

Study-note section DSA37

Ordinary array merging needs a linear auxiliary buffer.

Study-note section DSA37

There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA07

An n-level ordinary recursion with constant live storage per level needs Theta(n) stack space.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Data Structures & Algorithms practice questions · Question 27

Question 48 | Q00398

What is the tightest listed worst-case time bound for ordinary array mergesort that performs all merges and does not skip already ordered runs? Interpret logarithms explicitly as parenthesized in the options.

A. O(n log n)

B. $O(n^2)$

C. $O(n^2 \log n)$

D. $O(n (\log n)^2)$

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. O(n log n)

Ordinary mergesort splits the array into logarithmically many levels and performs linear total merging work per level. It therefore has n log n worst-case growth under the specified implementation. Sorted input does not skip any merges here. The quadratic and $n^2 \log n$ choices are looser upper bounds, while $n(\log n)^2$ grows faster than n log n. Asking for the tightest listed bound selects A uniquely.

Why each choice is right or wrong

A - Correct: This matches the actual n log n growth.

B - Incorrect: Quadratic is a valid but looser upper bound.

C - Incorrect: $n^2 \log n$ is a valid but looser upper bound.

D - Incorrect: $n(\log n)^2$ grows faster and is a looser upper bound.

Primary note proof

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA06

> For eventually nonnegative functions, f(n)=O(g(n)) means f(n)≤c g(n) for some positive c beyond a threshold k. Omega reverses the inequality; Theta requires both. Little-o means f/g tends to zero; little-omega means it grows without bound. These relations can describe best-case, average-case, worst-case or space functions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 38-44

> iv. Merge sort
> Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
> Algorithm:
> 1. Divide the array in to two halves.
> 2. Recursively sort the first n/2 items.
> 3. Recursively sort the last n/2 items.
> 4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 3 | lines 2-9

> I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:
> II. Big-Omega Notation (Ω): provides an asymptotic lower bound; f(n)= Ω(g (n)) means that the growth rate of f(n) is greater than or equal to g(n).
> III. Theta Notation (Θ): If f(n)= Θ (g(n)), then g(n) is an asymptotically tight bound for f(n) i.e f(n) and g(n) have the same rate of growth.

Question sources

Data Structures & Algorithms practice questions · Question 28

Question 49 | Q00399

What is the tightest listed best-case time bound for ordinary array mergesort that performs all merges and does not skip already ordered runs? Interpret logarithms explicitly as parenthesized in the options.

A. O(n log n)

B. $O(n^2)$

C. $O(n^2 \log n)$

D. $O(n (\log n)^2)$

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. O(n log n)

Ordinary mergesort splits the array into logarithmically many levels and performs linear total merging work per level. It therefore has n log n best-case growth under the specified implementation. Sorted input does not skip any merges here. The quadratic and $n^2 \log n$ choices are looser upper bounds, while $n(\log n)^2$ grows faster than n log n. Asking for the tightest listed bound selects A uniquely.

Why each choice is right or wrong

A - Correct: This matches the actual n log n growth.

B - Incorrect: Quadratic is a valid but looser upper bound.

C - Incorrect: $n^2 \log n$ is a valid but looser upper bound.

D - Incorrect: $n(\log n)^2$ grows faster and is a looser upper bound.

Primary note proof

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA06

> For eventually nonnegative functions, f(n)=O(g(n)) means f(n)≤c g(n) for some positive c beyond a threshold k. Omega reverses the inequality; Theta requires both. Little-o means f/g tends to zero; little-omega means it grows without bound. These relations can describe best-case, average-case, worst-case or space functions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Data Structures & Algorithms practice questions · Question 29

Question 50 | Q00400

Using the terminology of the course note, which listed phrase names the combining operation rather than a recognized organization or storage variant of mergesort?

A. in-place merge sort

B. bottom up merge sort

C. top down merge sort

D. linear merge sort

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: D. linear merge sort

Top-down mergesort recursively divides the sequence, while bottom-up mergesort iteratively combines runs of increasing widths. In-place variants change the storage tradeoff used while merging. Under the note's stated terminology, linear merge names the combining operation itself rather than a separate standardized sort variant. This is a distinction within the supplied naming convention, not a claim that no author could ever use that phrase differently.

Why each choice is right or wrong

A - Incorrect: In-place merging is a recognized storage variant.

B - Incorrect: Bottom-up mergesort is the iterative run-merging organization.

C - Incorrect: Top-down mergesort is the recursive organization.

D - Correct: Linear merge names the combining operation under the stated terminology.

Primary note proof

Study-note section DSA37

Top-down mergesort recurses; bottom-up merges runs of widths 1,2,4,... iteratively. In-place merging variants exist with different tradeoffs; linear merge names an operation, not a universally standardized sort variant. Internal sorting fits the working dataset in memory.

External sorting builds sorted runs and merges buffered streams from disk, aiming to reduce I/O. Reverse order alone does not characterize maximum merge comparisons.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Data Structures & Algorithms practice questions · Question 30

Question 51 | Q00401

Which of the following is not in place sorting algorithm by default?

A. merge sort

B. quick sort

C. heap sort

D. insertion sort

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: merge sort

The default merge algorithm explicitly uses an auxiliary array. The described partition, heap swaps and insertion can use the original array; recursion-stack use is distinct from requiring a second element array. Evidence: assigned PDF pages 11, 3.

Why each choice is right or wrong

A - Correct: Merge expressly allocates auxiliary array storage. See assigned PDF 11, 11, 11, 3.

B - Incorrect: Quicksort partitions in the existing array in its conventional form. See assigned PDF 11, 11, 11, 3.

C - Incorrect: Heap sorts by exchanging root and last within the array. See assigned PDF 11, 11, 11, 3.

D - Incorrect: Insertion rearranges the list in its existing slots. See assigned PDF 11, 11, 11, 3.

Primary note proof

Study-note section DSA37

Ordinary array merging needs a linear auxiliary buffer.

Study-note section DSA36

Quicksort normally rearranges in the same array but recursive storage can be linear in the worst case.

Study-note section DSA38

Building with bottom-up sinking is linear; repeated extraction gives O(n log n) worst time. Iterative sinking needs constant auxiliary storage. The method is generally unstable because long root-to-end exchanges can reorder equal keys. It is not uniquely the method for sorting by priorities: any comparison sort can compare a priority field. Heap extraction is its distinctive mechanism.

Study-note section DSA34

For each remaining item, save its value, shift larger preceding values right, and insert it into the gap.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which

each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Data Structures & Algorithms practice questions · Question 31

Question 52 | Q00402

Which of the following is not a stable sorting algorithm?

A. Quick sort

B. Cocktail sort

C. Bubble sort

D. Merge sort

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Standard quicksort is unstable.

The DSA31 sorting table identifies quicksort as unstable. Cocktail and bubble sorts preserve equal-key order when swapping only strict adjacent inversions. Ordinary merge sort preserves equal-key order when ties are taken from the left run first. Under these usual implementations, quicksort is the single listed algorithm that does not guarantee stability.

Why each choice is right or wrong

A - Correct: Standard quicksort partition exchanges can move equal keys past one another and do not guarantee stability.

B - Incorrect: Cocktail sort is stable with strict adjacent swaps, as described in DSA32.

C - Incorrect: Bubble sort preserves equal-key order when it swaps only strict adjacent inversions.

D - Incorrect: Merge sort preserves equal-key order by taking equal elements from the left run first.

Primary note proof

Study-note section DSA32

> Early stopping makes a sorted input linear: one no-swap pass finishes. A fixed pass schedule stays quadratic. Strict swaps preserve equal-key order. Cocktail sort runs bubble passes in both directions and is stable with strict swaps.

Study-note section DSA37

> With left-first tie handling, equal keys retain order.

Study-note section DSA31

> stack usually log n average, n worst; unstable

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

> ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
> Algorithm:
> 1. Choose a pivot value (mostly the first element is taken as the pivot value)
> 2. Position the pivot element and partition the list so that:
> • the left part has items less than or equal to the pivot value
> • the right part has items greater than or equal to the pivot value
> 3. Recursively sort the left part
> 4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 4 | lines 2-8

> C. Bubble Sort
> The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through

the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Data Structures & Algorithms practice questions · Question 32

Question 53 | Q00403

Which specified sorting implementations avoid recursion? A is ordinary recursive quicksort; B is top-down recursive mergesort; C uses iterative heap sinking; D is bottom-up mergesort. Select all that apply.

A. quick sort

B. top-down merge sort

C. heap sort with iterative sinking

D. bottom up merge sort

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C, D: C. heap sort with iterative sinking; D. bottom up merge sort

Bottom-up mergesort processes run widths 1, 2, 4 and so on in loops, without recursive splitting. Heapsort can also restore heap order with iterative sinking, making the specified C implementation nonrecursive. A and B are explicitly the recursive forms described in the note. The question fixes implementations because algorithm names alone do not force either recursive or iterative control flow.

Why each choice is right or wrong

A - Incorrect: The specified quicksort recursively sorts its partition ranges.

B - Incorrect: The specified top-down mergesort recursively sorts halves.

C - Correct: Iterative heap sinking requires no recursive calls.

D - Correct: Bottom-up run merging proceeds iteratively.

Primary note proof

Study-note section DSA37

> Top-down mergesort recurses; bottom-up merges runs of widths 1,2,4,... iteratively.

Study-note section DSA38

> Iterative sinking needs constant auxiliary storage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

> ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
> Algorithm:
> 1. Choose a pivot value (mostly the first element is taken as the pivot value)
> 2. Position the pivot element and partition the list so that:
> • the left part has items less than or equal to the pivot value
> • the right part has items greater than or equal to the pivot value
> 3. Recursively sort the left part
> 4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

> iv. Merge sort
> Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
> Algorithm:
> 1. Divide the array in to two halves.
> 2. Recursively sort the first n/2 items.
> 3. Recursively sort the last n/2 items.
> 4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 11 | lines 23-37

> iii. Heap Sort
> Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
> each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
> Algorithm:
> 1. Construct a binary tree
> • The root node corresponds to Data[0].
> • If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the

array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Question sources

Data Structures & Algorithms practice questions · Question 33

Question 54 | Q00404

What is the worst case time complexity of a quick sort algorithm?

A. O(N)

B. O(N log N)

C. O(N2)

D. O(log N)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: O(N2)

The source states O(n^2) in the worst case. Repeated extreme pivots leave subproblems n-1,n-2,..., whose partition work sums quadratically. Evidence: assigned PDF pages 11.

Why each choice is right or wrong

A - Incorrect: Linear does not include repeated near-full partitions. See assigned PDF 11.

B - Incorrect: n log n is the expected, not worst, cost. See assigned PDF 11.

C - Correct: Unbalanced partitions give quadratic work. See assigned PDF 11.

D - Incorrect: Logarithmic describes neither partition scans nor their total. See assigned PDF 11.

Primary note proof

Study-note section DSA36

Choose a pivot, partition, then recursively sort the pieces. Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work. Expected n log n needs an input/random-pivot model. Random and median-of-three choices have different tradeoffs; neither is universally best.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Question sources

Data Structures & Algorithms practice questions · Question 34

Question 55 | Q00405

For a strictly increasing array of at least five distinct keys, which listed pivot rule is deterministic and avoids choosing an extreme key on the first partition? Interpret median-of-three as the median of the first, middle and last keys.

A. first element

B. last element

C. median-of-three partitioning

D. random element

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: C. median-of-three partitioning

On a strictly increasing array, the first key is the minimum and the last key is the maximum, so those deterministic endpoint rules choose extremes. The median of the first, middle and last keys is the middle-position value, which is neither extreme for the stated size. A random element is not a deterministic rule and can choose an endpoint. This criterion selects C without claiming a universally best practical pivot strategy.

Why each choice is right or wrong

A - Incorrect: The first key is the minimum on this input.

B - Incorrect: The last key is the maximum on this input.

C - Correct: The sample median is the middle-position value and is not extreme.

D - Incorrect: Random selection is not deterministic and may choose an extreme.

Primary note proof

Study-note section DSA36

Choose a pivot, partition, then recursively sort the pieces. Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work. Expected n log n needs an input/random-pivot model. Random and median-of-three choices have different tradeoffs; neither is universally best.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Question sources

Data Structures & Algorithms practice questions · Question 35

Question 56 | Q00406

What is the average running time of a quick sort algorithm?

A. O(N2)

B. O(N)

C. O(N log N)

D. O(log N)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer C: O(N log N)

Expected O(n log n) is explicitly given. Under ordinary nonadversarial/random-pivot assumptions this is the average runtime intended by the question. Evidence: assigned PDF pages 11.

Why each choice is right or wrong

A - Incorrect: Quadratic is the stated worst case. See assigned PDF 11.

B - Incorrect: Linear omits recursive partition levels. See assigned PDF 11.

C - Correct: Matches the stated expected n log n growth. See assigned PDF 11.

D - Incorrect: Logarithmic omits linear partition work. See assigned PDF 11.

Primary note proof

Study-note section DSA36

Choose a pivot, partition, then recursively sort the pieces. Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work. Expected n log n needs an input/random-pivot model. Random and median-of-three choices have different tradeoffs; neither is universally best.

Study-note section DSA-S01

Fix the input size before comparing cases. Best-case cost is the minimum resource use over legal inputs of that size; worst-case cost is the maximum. Average-case cost averages costs using a stated input distribution. These three input-case functions can each be described using upper, lower or tight asymptotic bounds; the case name does not select the bound notation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Question sources

Data Structures & Algorithms practice questions · Question 36

Question 57 | Q00407

Which option gives the most complete set of the listed situations in which linear search can be an appropriate direct method? Select the combined answer rather than only one valid situation.

A. When the list has only a few elements

B. When performing a single search in an unordered list

C. Used all the time

D. When the list has only a few elements and When performing a single search in an unordered list

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: Both short lists and a single unsorted-list query favor scanning.

Linear scanning can suit a short list because setup costs may dominate, and it can answer a single unsorted query without first building an ordered index or sorting the data. These are two supported use cases, not conditions that must occur simultaneously. A and B each give one true but incomplete situation. D combines both, while the universal claim that scanning should be used all the time is false.

Why each choice is right or wrong

A - Incorrect: True but incomplete: a short list is one supported case, but this omits the single unsorted query.

B - Incorrect: True but incomplete: a single unsorted query is supported, but this omits the short-list case.

C - Incorrect: Linear search is not preferable for every workload or available representation.

D - Correct: The combined answer includes both supported situations.

Primary note proof

Study-note section DSA28

> For a single unsorted query, sorting first is rarely justified. Small lists may favour scanning because setup matters. Sorting does not avoid examining all values for mean, variance or arbitrary parity counts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

> Linear/Sequential searching
> • Works with unsorted List
> • Loop through the list starting at the first element
> • Time is proportional to the size of input (n) and we call this time complexity O(n).

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Question sources

Data Structures & Algorithms practice questions · Question 37

Question 58 | Q00408

What is the best case for linear search?

A. O(nlogn)

B. O(logn)

C. O(n)

D. O(1)

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: O(1)

An immediate match at the first list element takes one comparison. Treat complexity options as the tight order. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: n log n is not the immediate-match cost. See assigned PDF 3.

B - Incorrect: Logarithmic is not needed for the first inspected item. See assigned PDF 3.

C - Incorrect: Linear is the complete-scan order. See assigned PDF 3.

D - Correct: The first-item match requires constant work. See assigned PDF 3.

Primary note proof

Study-note section DSA28

> Compare the target with successive elements until equal or exhausted. Unsorted data is allowed; sorted data is allowed too. A first-element match costs Theta(1). An absent target or last-position match needs n comparisons, giving Theta(n) worst time. Uniformly likely successful positions average (n+1)/2 comparisons, still linear.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

> Linear/Sequential searching
> • Works with unsorted List
> • Loop through the list starting at the first
> element
> • Time is proportional to the size of input

(n) and we call this time complexity
O(n).

Question sources

Data Structures & Algorithms practice questions · Question 38

Question 59 | Q00409

What is the worst case for linear search?

A. O(nlogn)

B. O(logn)

C. O(n)

D. O(1)

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: O(n)

An absent target or a match at the final position requires inspecting n elements, giving tight linear worst-case growth. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: n log n is a loose bound, not the scan order. See assigned PDF 3.

B - Incorrect: Logarithmic cannot inspect all n elements. See assigned PDF 3.

C - Correct: A full scan is proportional to n. See assigned PDF 3.

D - Incorrect: Constant work describes immediate success, not worst case. See assigned PDF 3.

Primary note proof

Study-note section DSA28

Compare the target with successive elements until equal or exhausted. Unsorted data is allowed; sorted data is allowed too. A first-element match costs Theta(1). An absent target or last-position match needs n comparisons, giving Theta(n) worst time. Uniformly likely successful positions average (n+1)/2 comparisons, still linear.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List

• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Data Structures & Algorithms practice questions · Question 39

Question 60 | Q00410

What is the worst case complexity of binary search using recursion?

A. O(nlogn)

B. O(logn)

C. O(n)

D. O(n2)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer B: O(logn)

Each recursive step halves the candidate interval; after k steps its size is at most n/2^k. Reaching size one requires about log2 n steps. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: n log n overcounts a single search path. See assigned PDF 3.

B - Correct: Repeated halving gives logarithmic search depth. See assigned PDF 3.

C - Incorrect: Linear ignores the discarded half at every step. See assigned PDF 3.

D - Incorrect: Quadratic has no corresponding nested scan. See assigned PDF 3.

Primary note proof

Study-note section DSA29

On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Study-note section DSA29

Worst probes are floor($\log_2 n$)+1 for n>0: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal

loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Question sources

Data Structures & Algorithms practice questions · Question 40

Question 61 | Q00411

For recursive binary search on a sorted random-access array of n distinct keys, each stored key is equally likely to be the successful target. With constant-time comparisons, what is the tightest listed average-time upper bound?

A. O(nlogn)

B. O(logn)

C. O(n)

D. O(n2)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer B: B. O(logn)

Binary search repeatedly halves the remaining interval. Under equally likely successful keys, only a limited number can be found at shallow decision-tree levels, while a constant fraction require logarithmic depth. Together with the logarithmic upper bound, this gives Theta(log n) average probes. The larger displayed upper bounds are valid but loose; the requested tightest listed bound is therefore O(log n).

Why each choice is right or wrong

A - Incorrect: n log n is a looser upper bound on logarithmic average work.

B - Correct: log n is the tight average order under the stated successful-target distribution.

C - Incorrect: Linear is a valid but looser upper bound.

D - Incorrect: Quadratic is a valid but much looser upper bound.

Primary note proof

Study-note section DSA-S01

> For successful binary search with all stored positions equally likely, average probes have order Theta(log n). The probe decisions form a binary tree: only a bounded number of positions occur in shallow levels, while a constant fraction require logarithmic depth. Interval halving also gives the logarithmic upper bound. A distribution concentrating on an immediate midpoint hit would give a different average.

Study-note section DSA29

> On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Question sources

Data Structures & Algorithms practice questions · Question 41

Question 62 | Q00412

Which task cannot directly use ordinary binary interval elimination under its stated conditions? The interval-union subtask in B uses a sorted array of interval endpoints; C has an ordered monotone good/bad boundary.

A. To find the lower/upper bound in an ordered sequence

B. Finding endpoint bounds in a sorted endpoint array while forming an interval union

C. Finding a monotone good/bad boundary during debugging

D. Directly searching an unordered list without sorting or an ordered index

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: D. Directly searching an unordered list without sorting or an ordered index

Binary search needs ordering that justifies rejecting one half of its candidate interval. Lower and upper bounds in an ordered sequence have that property. Searching sorted interval endpoints during an interval-union task also supplies an ordered subproblem, and a monotone debugging boundary permits the same halving logic. Direct searching in an unordered list supplies no such invariant, so a midpoint comparison cannot safely discard a half.

Why each choice is right or wrong

A - Incorrect: Boundary variants operate on the ordered candidate sequence.

B - Incorrect: The specified endpoint lookup is an ordered binary-search subtask.

C - Incorrect: The specified monotone good/bad boundary supports binary elimination.

D - Correct: An unordered list lacks the order needed to safely discard half after comparison.

Primary note proof

Study-note section DSA29

> On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Study-note section DSA29

> Binary debugging needs an ordered good/bad boundary, not arbitrary fault locations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Question sources

Data Structures & Algorithms practice questions · Question 42

Question 63 | Q00413

Which standard linear structure is defined by a chain of nodes, each containing a payload and one successor link, with a null successor ending an ordinary noncircular chain?

A. linked list

B. Array

C. Binary tree

D. None of these

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. linked list

The payload stores an item and the successor link connects its node to the next item's node. Following those links produces a linear sequence ending at null, which is the ordinary singly linked-list representation. An array instead reaches equal-sized positions by address arithmetic, and a binary tree has a branching parent/child contract rather than this one-successor sequence contract. The linked-list option is therefore the matching standard name.

Why each choice is right or wrong

A - Correct: A linked list represents its sequence through payload-plus-successor nodes.

B - Incorrect: An array uses indexed positions rather than this successor-chain representation.

C - Incorrect: A binary tree specifies up to two child branches rather than the stated linear successor contract.

D - Incorrect: The linked-list option supplies the matching structure.

Primary note proof

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain.

Study-note section DSA02

> For equal-size elements, address(A[i]) = base + i × element_size under zero-based indexing.

Study-note section DSA18

> Binary means at most two children.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Question sources

Data Structures & Algorithms practice questions · Question 43

Question 64 | Q00414

What is the time complexity to count the number of elements in the linked list?

A. O(1)

B. O(n)

C. O(logn)

D. None of the mentioned

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer B: O(n)

Without a stored count, follow next pointers from start to NULL and increment once per visited node. Exactly n node visits are needed. Evidence: assigned PDF pages 5.

Why each choice is right or wrong

A - Incorrect: Constant time requires an additional maintained count. See assigned PDF 5.

B - Correct: Traversal visits every node once. See assigned PDF 5.

C - Incorrect: No halving shortcut exists in the next-pointer chain. See assigned PDF 5.

D - Incorrect: B supplies the traversal order. See assigned PDF 5.

Primary note proof

Study-note section DSA08

To display or count nodes, follow next from head until null, processing each node. Counting is Theta(n) unless size is maintained. An array-linked list still follows links unless separately position-indexed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 5 | lines 26-34

III. Displaying the list of Nodes
After adding nodes to the list, we need to display the list of nodes on the screen. Here are the steps required to display all nodes item in the list:
1. Set a temporary pointer to point to the same thing as the start pointer.
2. If the pointer points to NULL, display the message "End of list" and stop.
3. Otherwise, display the details of the node pointed to by the start pointer.
4. Make the temporary pointer point to the same thing as the next pointer of the node it is currently indicating.
5. Jump back to step 2.

Question sources

Data Structures & Algorithms practice questions · Question 44

Question 65 | Q00415

Linked lists are not suitable to for the implementation of?

A. Insertion sort

B. Radix sort

C. Polynomial manipulation

D. Binary search

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: Binary search loses its usual advantage on linked lists.

DSA29 explains that linked midpoint traversal loses logarithmic total time. That makes an ordinary linked list unsuitable for the usual efficient binary-search implementation, although an algorithm can still be written. The other tasks do not require constant-time midpoint access: insertion sort can relink nodes, polynomial terms can occupy nodes, and radix digit buckets can retain stable node sequences.

Why each choice is right or wrong

A - Incorrect: Insertion sorting can find an insertion location and relink nodes without requiring constant-time midpoint access.

B - Incorrect: Radix digit buckets can preserve node order across passes without a random-access midpoint operation.

C - Incorrect: Polynomial terms are explicitly listed as a linked-list application in DSA10.

D - Correct: Traversing to linked-list midpoints defeats the standard logarithmic total-time advantage of binary search.

Primary note proof

Study-note section DSA29

> Linked midpoint traversal loses logarithmic total time.

Study-note section DSA34

> Few inversions mean few shifts, favouring insertion for nearly sorted or small arrays. Shift only strictly larger keys for stability. A linked implementation saves array shifts but still must locate each insertion position.

Study-note section DSA10

> Lists suit changing sequences when references to insertion positions are already available. Finding a position is still linear. Lists can hold polynomial terms (coefficient, exponent), directory or block links, child lists in nonbinary trees, and hash chains. Browser history can use a doubly linked cursor. A playlist may be circular; airline seat lookup requires a specified seat-record model, not an automatic linked-list answer.

Study-note section DSA39

> Stable intermediate passes preserve the ordering already established by less-significant digits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
> to the chain.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Question sources

Data Structures & Algorithms practice questions · Question 45

Question 66 | Q00416

Which of these is an application of linked lists?

A. To implement file systems

B. For separate chaining in hash-tables

C. To implement non-binary trees

D. All of the mentioned

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: All listed applications.

DSA10 explicitly lists directory or block links, hash chains, and nonbinary-tree child lists as applications of linked nodes. These correspond to the three preceding options. Therefore All of the mentioned is the complete answer. The statement says linked lists can implement these components; it does not require every file system or tree representation to use them.

Why each choice is right or wrong

A - Incorrect: File-system directory or block links are valid uses, but this option omits the other supported applications.

B - Incorrect: Separate chaining uses linked bucket entries, but this is only one of the supported applications.

C - Incorrect: Child lists can represent nonbinary-tree children, alongside the other listed uses.

D - Correct: All three preceding options correspond to linked-record applications explicitly listed in DSA10.

Primary note proof

Study-note section DSA10

Lists suit changing sequences when references to insertion positions are already available. Finding a position is still linear. Lists can hold polynomial terms (coefficient, exponent), directory or block links, child lists in nonbinary trees, and hash chains. Browser history can use a doubly linked cursor. A playlist may be circular; airline seat lookup requires a specified seat-record model, not an automatic linked-list answer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Question sources

Data Structures & Algorithms practice questions · Question 46

Question 67 | Q00417

In linked list each node contain minimum of two fields. One field is data field to store the data second field is?

A. Pointer to character

B. Pointer to integer

C. Pointer to node

D. Node

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Pointer to node

The second minimum field points to the next node of the same node structure, alongside the payload. Evidence: assigned PDF pages 4.

Why each choice is right or wrong

A - Incorrect: A character pointer is not the next-node link unless the entire node type were a character, which it is not. See assigned PDF 4.

B - Incorrect: An integer pointer cannot stand for the described structured node link. See assigned PDF 4.

C - Correct: The link points to a node of the list. See assigned PDF 4.

D - Incorrect: The field is a pointer to a node, not a nested node stored by value. See assigned PDF 4.

Primary note proof

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
> to the chain.

Question sources

Data Structures & Algorithms practice questions · Question 47

Question 68 | Q00418

For repeated queries for the item at logical position k, which listed representation gives constant-time lookup? A-C maintain ordinary links and endpoints only; D additionally maintains a random-access array mapping each logical position to its node. Ignore the cost of maintaining this index during updates.

A. Singly linked list

B. Doubly linked list

C. Circular linked list

D. Array implementation with a separate logical-position-to-node index

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer D: D. Array implementation with a separate logical-position-to-node index

Ordinary singly, doubly and circular linked lists must follow links to reach a logical position, giving linear worst-case positional access. Storing nodes in an array alone does not change that. The explicit extra index in D is different: it maps each logical position directly to the corresponding node, so a random-access array lookup answers the query in constant time. Update costs are excluded by the stated workload.

Why each choice is right or wrong

A - Incorrect: Ordinary next-link traversal is linear in the worst case.

B - Incorrect: Using either endpoint still requires linear worst-case traversal.

C - Incorrect: Circular links do not supply a direct logical-position index.

D - Correct: The maintained position-to-node array permits direct indexed lookup.

Primary note proof

Study-note section DSA08

> To display or count nodes, follow next from head until null, processing each node. Counting is Theta(n) unless size is maintained. An array-linked list still follows links unless separately position-indexed.

Study-note section DSA02

> For equal-size elements, address(A[i]) = base + i × element_size under zero-based indexing. This arithmetic gives constant-time positional access in the usual RAM cost model. A row-major rectangular matrix uses base + (row × columns + column) × element_size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 5 | lines 26-34

III. Displaying the list of Nodes
After adding nodes to the list, we need to display the list of nodes on the screen. Here are the steps required to display all nodes item in the list:
1. Set a temporary pointer to point to the same thing as the start pointer.
2. If the pointer points to NULL, display the message "End of list" and stop.
3. Otherwise, display the details of the node pointed to by the start pointer.
4. Make the temporary pointer point to the same thing as the next pointer of the node it is currently indicating.
5. Jump back to step 2.

Question sources

Data Structures & Algorithms practice questions · Question 48

Question 69 | Q00419

Linked list is considered as an example of _____ type of memory allocation.

A. Dynamic

B. Static

C. Compile time

D. None of the mentioned

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Dynamic

Nodes are allocated with new as they are needed; the source explicitly calls linked-list stack memory dynamic. Evidence: assigned PDF pages 5, 6.

Why each choice is right or wrong

A - Correct: Node allocation occurs during execution. See assigned PDF 5, 6.

B - Incorrect: Static allocation is the contrasting array implementation. See assigned PDF 5, 6.

C - Incorrect: Compile-time allocation does not describe new-node creation. See assigned PDF 5, 6.

D - Incorrect: Dynamic is explicitly supported. See assigned PDF 5, 6.

Primary note proof

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Study-note section DSA03

> In C++, match new with delete and new[] with delete[].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 5 | lines 3-8

> While adding a node to a list, the chain needs to be rearranged properly so that the each nodes are connected to each other. It is possible to add a node anywhere in the chain (at the start, middle or end). Let’s consider adding a node at the end of the list:
> Step 1: First we need to reserve a space for the new node using the keyword new.
> Node *temp = new *Node; // Reserving a space for temporary pointer which points to new node;

Data Structure and Algorithms module | PDF page 6 | lines 43-45

> II. Linked list (dynamic memory)
> The linked list implementation of stack is adding items at the end of the chain (Push item) and deleting the end node of the list (Pop item).

Question sources

Data Structures & Algorithms practice questions · Question 49

Question 70 | Q00420

In Linked List implementation, a node carries information regarding

A. Data

B. Link

C. Data and Link

D. None of the mentioned

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Data and Link

A node combines its payload with a next-node pointer. Neither field alone represents the complete linked-list node. Evidence: assigned PDF pages 4.

Why each choice is right or wrong

A - Incorrect: Data alone omits the next link. See assigned PDF 4.

B - Incorrect: A link alone omits the payload. See assigned PDF 4.

C - Correct: Both data and link are explicitly required. See assigned PDF 4.

D - Incorrect: C matches the stated two-field structure. See assigned PDF 4.

Primary note proof

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
> to the chain.

Question sources

Data Structures & Algorithms practice questions · Question 50

Question 71 | Q00421

Process of inserting an element in stack is called _____.

A. Create

B. Push

C. Evaluation

D. Pop

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: Push

The module defines push as adding an element to the stack top. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Incorrect: Create would initialize a structure, not name its insertion operation. See assigned PDF 6.

B - Correct: Push adds an element. See assigned PDF 6.

C - Incorrect: Evaluation is an application rather than insertion. See assigned PDF 6.

D - Incorrect: Pop removes an element. See assigned PDF 6.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Data Structures & Algorithms practice questions · Question 51

Question 72 | Q00422

In a stack, if a user tries to remove an element from empty stack it is called _____.

A. Underflow

B. Empty collection

C. Overflow

D. Garbage Collection

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Underflow

The stack pop code first tests empty at top==0. Removing from an empty structure is the underflow condition, explicitly named for dequeue and applied here to the same empty-removal failure. Evidence: assigned PDF pages 6, 7.

Why each choice is right or wrong

A - Correct: Underflow denotes attempting removal without an available element. See assigned PDF 6, 7.

B - Incorrect: Empty collection describes the state rather than the operation failure. See assigned PDF 6, 7.

C - Incorrect: Overflow concerns adding to a full structure. See assigned PDF 6, 7.

D - Incorrect: Garbage collection concerns reclaiming storage, not empty pop. See assigned PDF 6, 7.

Primary note proof

Study-note section DSA11

> Overflow is an attempted insertion into full storage; underflow is an attempted removal from empty storage. Merely checking isEmpty does not underflow.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 24-42

> I. Array implementation (static memory).
> Push(n): Adding item at top
> top is the current top of stack and n is its
> maximum size
> Begin
> if (top == n) then stack is full;
> else

```
top := top+1;
stack[top] := item;
end:
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;
```

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Data Structures & Algorithms practice questions · Question 52

Question 73 | Q00423

Process of removing an element from stack is called _____

A. Create

B. Push

C. Evaluation

D. Pop

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: Pop

The source names removing an element Pop and adding one Push. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Incorrect: Create does not remove the stack top. See assigned PDF 6.

B - Incorrect: Push adds rather than removes. See assigned PDF 6.

C - Incorrect: Evaluation uses stack operations but is not their deletion name. See assigned PDF 6.

D - Correct: Pop removes the current top. See assigned PDF 6.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Data Structures & Algorithms practice questions · Question 53

Question 74 | Q00424

A serial-storage interface restricts insertion and removal to the same current end: an item must be removed before any item stored earlier beneath it can be removed. Which ADT matches this restricted access policy?

A. Heap

B. Binary Tree

C. Array

D. Stack

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: D. Stack

The restriction makes the latest inserted item the next removable item. That is the stack's LIFO policy, with both push and pop acting at the top. Serial access alone would not establish this rule, so the same-end removal restriction is stated explicitly. A heap selects by priority, a binary tree supplies branching relationships, and an ordinary array permits indexed access rather than imposing this removal policy.

Why each choice is right or wrong

A - Incorrect: A heap selects an extreme priority rather than the latest insertion.

B - Incorrect: A binary tree supplies hierarchy rather than this same-end policy.

C - Incorrect: An array does not inherently impose this removal order.

D - Correct: A stack's push/pop policy matches the specified LIFO restriction.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA45

> ADTs do not force one representation, and serial-access memory is not inherently LIFO.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> "A stack is an ordered list in which all insertions and deletions are made at one end, called the top". Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Data Structures & Algorithms practice questions · Question 54

Question 75 | Q00425

A singly linked queue maintains only its front pointer and owns all its nodes. Emptying means immediately visiting and freeing every node. Which option gives the complete set of listed operations with tight worst-case linear time for n nodes?

A. Insertion

B. Deletion

C. To empty a queue

D. Both Insertion and To empty a queue

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer D: D. Both Insertion and To empty a queue

Enqueue must traverse from the front to find the tail because no rear pointer is maintained. Emptying the owned queue must also visit and release all n nodes under the specified reclamation contract. Deleting one front node requires only a constant number of link and deallocation operations. Therefore both insertion and complete emptying are linear, and the aggregate option D is the complete selection.

Why each choice is right or wrong

A - Incorrect: Insertion is linear, but this omits the also-linear emptying operation.

B - Incorrect: A single front deletion is constant time.

C - Incorrect: Emptying is linear, but this omits insertion.

D - Correct: Both insertion and immediate full-node cleanup are linear.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Study-note section DSA09

> Deleting the tail of a singly linked list requires its predecessor, even when tail itself is known. Handle empty and single-node lists separately; after removing the only node, set both head and tail to null. To remove a known successor, bypass it before deallocation. Clearing an owned n-node chain visits and releases every node.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 5 | lines 13-24

Step 3: Then add the node at the end of the list. For this we need to check whether the list is empty or not. If list is empty the new node will be the first (Start) node, otherwise navigate forward starting form the first node till last node found and add it next to this node which make the new node the last node.
if (start == NULL) // If list is empty, then new node become start node
start = temp;
else {
temp2 = start; // list not empty! Thus temp2 starts from first node
while (temp2->next!= NULL) { // till the current last node found navigates forward
temp2 = temp2->next; // Move to next link in chain
}
temp2->next = temp; // new node (temp) appended at the end of the list

Data Structure and Algorithms module | PDF page 8 | lines 35-37

III. Linked list implementation of enqueue and dequeue operations
Enqueue- is inserting a node at the end of a linked list
Dequeue- is deleting the first node in the list

Data Structure and Algorithms module | PDF page 5 | lines 35-41

IV. Deleting a node from the list
When a node is deleted, the space that it took up should be reclaimed. Otherwise the computer will eventually run out of memory space. This is done with the delete instruction:
delete temp; // Release the memory pointed to by temp
However, we can't just delete the nodes willy-nilly as it would break the chain. We need to reassign the pointers and then delete the node at the last moment. We can delete nodes at the start, middle or end of the list. Here let’s see the steps to delete at the end.

Question sources

Data Structures & Algorithms practice questions · Question 55

Question 76 | Q00426

A linear list of elements in which deletion can be done from one end (front) and insertion can take place only at the other end (rear) is known as a ?

A. Queue

B. Stack

C. Tree

D. Linked list

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Queue

The queue definition makes insertion rear-only and deletion front-only. A linked list can implement that policy, but does not by itself impose the two restrictions. Evidence: assigned PDF pages 7, 6, 4.

Why each choice is right or wrong

A - Correct: Matches the defined queue interface. See assigned PDF 7, 6, 4.

B - Incorrect: Stack uses one common endpoint for both operations. See assigned PDF 7, 6, 4.

C - Incorrect: Tree operations are not front/rear list operations. See assigned PDF 7, 6, 4.

D - Incorrect: A generic linked list permits operations beyond queue restrictions. See assigned PDF 7, 6, 4.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem

• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Question sources

Data Structures & Algorithms practice questions · Question 56

Question 77 | Q00427

In Breadth First Search of Graph, which of the following data structure is used?

A. Stack

B. Queue

C. Linked list

D. None of the mentioned

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: Queue

DSA41 repeatedly dequeues a vertex and enqueues undiscovered neighbors. FIFO service preserves the order of distance layers. Queue is therefore the standard BFS frontier structure. A linked list may store the queue but does not alone define this ordering; a stack chooses latest pending work and instead supports the depth-first discipline.

Why each choice is right or wrong

A - Incorrect: A stack selects the latest pending vertex and does not maintain breadth-first distance layers.

B - Correct: A FIFO queue processes earlier discoveries before later layers, matching standard breadth-first search.

C - Incorrect: Linked nodes can store a queue, but linked-list structure alone specifies no BFS frontier policy.

D - Incorrect: Queue is offered and directly matches the algorithm given in DSA41.

Primary note proof

Study-note section DSA41

> Mark the source when enqueuing it. Repeatedly dequeue a vertex, examine its neighbours, and enqueue each unvisited neighbour while marking it. Store predecessor[v] when discovering v if a path must be reconstructed. FIFO ensures vertices at distance d are explored before later distance layers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Data Structures & Algorithms practice questions · Question 57

Question 78 | Q00428

A data structure in which elements can be inserted or deleted at/from both the ends but not in the middle is?

A. Queue

B. Circular queue

C. Dequeue

D. Priority queue

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Dequeue

Interpret the option spelling Dequeue as deque, the double-ended queue defined on page 8. That structure allows insertion and deletion at either end. Evidence: assigned PDF pages 8, 7.

Why each choice is right or wrong

A - Incorrect: An ordinary queue restricts insertion and deletion to different fixed ends. See assigned PDF 8, 7.

B - Incorrect: Circular storage changes reuse, not the ordinary queue interface. See assigned PDF 8, 7.

C - Correct: Deque is explicitly double-ended; the option has a terminology typo. See assigned PDF 8, 7.

D - Incorrect: Priority queue chooses service by key, not arbitrary endpoint operations. See assigned PDF 8, 7.

Primary note proof

Study-note section DSA16

> A deque permits both-end insertion/removal without guaranteeing middle access. Endpoint-linked operations are constant but middle indexing is linear. A circular-array deque may support indexed access. Dequeue names removal; deque names the structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 38-40

> Deque (pronounced as Deck): is a Double Ended Queue in which insertion and deletion can occur at either end.
> Priority Queue:- is a queue where each data has an associated key provided at insertion time.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Data Structures & Algorithms practice questions · Question 58

Question 79 | Q00429

Queues serve major role in

A. Simulation of recursion

B. Simulation of arbitrary linked list

C. Simulation of limited resource allocation

D. All of the mentioned

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Simulation of limited resource allocation

The module lists print jobs, disk requests and waiting lines: all allocate a limited resource among waiting requests. Recursion is expressly a stack use, so all is false. Evidence: assigned PDF pages 8, 6.

Why each choice is right or wrong

A - Incorrect: Recursion is listed under stack applications. See assigned PDF 8, 6.

B - Incorrect: Arbitrary linked-list simulation is not a stated queue role. See assigned PDF 8, 6.

C - Correct: Waiting requests for shared resources fit the listed applications. See assigned PDF 8, 6.

D - Incorrect: A is false, so the conjunction cannot hold. See assigned PDF 8, 6.

Primary note proof

Study-note section DSA14

> FIFO suits equal-priority printer jobs, shared-resource waiting, asynchronous buffers and level-order exploration. A play sequence can also use a queue. Distinguish airline seat records from waiting requests; an application name alone does not force one representation.

Study-note section DSA12

> Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 41-48

> Application of Queue
> Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
> • Print server- maintains a queue of print jobs
> • Disk Driver- maintains a queue of disk input/output requests

• Task scheduler in multiprocessing system- maintains priority queues of processes
• Telephone calls in a busy environment -maintains a queue of telephone calls
• Simulation of waiting line- models real life queues (e.g. supermarkets checkout, customer service line (Bank, custom, airport etc))

Data Structure and Algorithms module | PDF page 6 | lines 46-52

Applications of Stack
• It is very useful to evaluate arithmetic expressions. (Postfix Expressions)
• Infix to Postfix Transformation
• It is useful during the execution of recursive programs
• A Stack is useful for designing the compiler in operating system to store local variables inside a function block.
• A stack (memory stack) can be used in function calls including recursion.

Question sources

Data Structures & Algorithms practice questions · Question 59

Question 80 | Q00430

Which listed service discipline is a stack policy rather than FIFO, a circular FIFO representation, or priority-based queue service? Interpret B as insertion and removal at the same top end.

A. Ordinary queue

B. Single-ended insertion and removal at the same top (LIFO)

C. Circular queue

D. Priority queue

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: B. Single-ended insertion and removal at the same top (LIFO)

An ordinary queue serves front elements after rear insertion, while a circular queue changes how its FIFO storage slots are reused. A priority queue chooses according to a priority key. Inserting and removing at the same top end instead gives LIFO stack service. Explicitly defining the formerly vague single-ended phrase makes this a policy distinction rather than an unsupported claim that a particular phrase is forbidden terminology.

Why each choice is right or wrong

A - Incorrect: Ordinary queue service is FIFO.

B - Correct: Same-top insertion/removal is LIFO stack service.

C - Incorrect: Circular storage can implement FIFO queue service.

D - Incorrect: Priority queue service selects by an extreme priority key.

Primary note proof

Study-note section DSA11

Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA14

An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Study-note section DSA17

A priority queue returns the smallest or largest priority key under its convention. Urgent patients may precede earlier arrivals. Insertions and extractions can interleave. Binary heaps support constant peek and logarithmic insert/extract.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

“A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 7 | lines 24-26

II. Circular array implementation of Queue Operations
To overcome the above drawback of simple array implementation a circular array can be used, in
which freed spaces are re-used to store data.

Data Structure and Algorithms module | PDF page 8 | lines 40-45

Priority Queue:- is a queue where each data has an associated key provided at insertion time.
Application of Queue
Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
• Print server- maintains a queue of print jobs
• Disk Driver- maintains a queue of disk input/output requests
• Task scheduler in multiprocessing system- maintains priority queues of processes

Question sources

Data Structures & Algorithms practice questions · Question 60

Question 81 | Q00543

Which claims contradict the ordinary unique-key map contract? Select all that apply.

A. Allow duplicate keys

B. Cannot be traversed

C. Allow duplicate values

D. None of the above

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A, B: A. Allow duplicate keys; B. Cannot be traversed

A map associates each unique key with a value. Different keys may share the same value, and its entries can be traversed even when a particular iteration order is not promised. Thus allowing duplicate keys and forbidding all traversal both contradict the contract. Allowing duplicate values does not contradict it. Since contradictory claims exist, the none-of-the-above option is not selected.

Why each choice is right or wrong

A - Correct: Duplicate keys contradict the unique-key map contract.

B - Correct: Map entries can be traversed, so this prohibition is false.

C - Incorrect: Different keys may share values, making this a valid property.

D - Incorrect: A and B do contradict the contract.

Primary note proof

Study-note section DSA27

> A map associates unique keys with values; different keys may share values, and entries can be traversed. A set forbids duplicate elements. Null and iteration-order policies depend on implementation. Lists preserve positions and generally allow duplicates; Java LinkedList also permits null.

Question sources

Computer Science model exam · 2015 · Question 12

Computer Science model exam · 2015 · Question 113

Question 82 | Q00591

Consider a set implementation that permits one null element and does not promise an iteration order. Which statement about this implementation is incorrect?

A. Unordered collection

B. Duplicates are not allowed

C. null is not allowed

D. None of the above

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: C. null is not allowed

All sets forbid duplicate elements, but null acceptance and iteration order depend on the implementation. The implementation explicitly permits null and promises no iteration order. Under those policies, calling its collection unordered is consistent and rejecting duplicates remains correct. The claim that null is not allowed is the one contradiction. The none-of-the-above alternative fails because that contradictory claim is present.

Why each choice is right or wrong

A - Incorrect: No iteration order is promised under the specified policy.

B - Incorrect: Rejecting duplicate elements is the set contract.

C - Correct: The stated implementation permits null, contradicting this prohibition.

D - Incorrect: C is incorrect, so none-of-the-above fails.

Primary note proof

Study-note section DSA27

> A map associates unique keys with values; different keys may share values, and entries can be traversed. A set forbids duplicate elements. Null and iteration-order policies depend on implementation. Lists preserve positions and generally allow duplicates; Java LinkedList also permits null.

Question sources

Computer Science model exam · 2015 · Question 60

Question 83 | Q00593

Which one of the following is correct about list data structure?

A. Unordered collection

B. Duplicates are not allowed

C. null is not allowed

D. None of the above

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer D: None of the above under the ordinary list contract.

DSA27 says lists preserve positions and generally allow duplicates; it also gives Java LinkedList as a list permitting null. Therefore Unordered collection and Duplicates are not allowed misdescribe the ordinary list contract. The universal null prohibition is refuted by the supplied example. None of those statements is a generally correct list property, so D is supported.

Why each choice is right or wrong

A - Incorrect: Lists preserve positions, so calling the ordinary list contract unordered confuses positional order with sortedness.

B - Incorrect: Ordinary lists allow repeated elements; duplicate rejection belongs to sets rather than the general list contract.

C - Incorrect: Java LinkedList explicitly permits null, refuting a universal null prohibition for list structures.

D - Correct: None of the preceding statements describes a generally valid property of the ordinary list contract.

Primary note proof

Study-note section DSA27

> A map associates unique keys with values; different keys may share values, and entries can be traversed. A set forbids duplicate elements. Null and iteration-order policies depend on implementation. Lists preserve positions and generally allow duplicates; Java LinkedList also permits null.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and

pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Question sources

Computer Science model exam · 2015 · Question 62

Question 84 | Q00711

In a binary search tree that stores distinct keys with strict left-smaller/right-larger ordering, which statements are true? Select all that apply.

A. Every node has at most two children.

B. The value on left sub tree of the binary search tree can be equal to the root node value.

C. Every node stores a unique key value.

D. All of the above

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A, C: A. Every node has at most two children.; C. Every node stores a unique key value.

Binary limits a node to at most two children; it does not require every node to have exactly two. The specified distinct-key model also requires uniqueness. Strict subtree ordering places every left-subtree key below its ancestor, so an equal key cannot appear there. A and C consequently hold under the stated model, while B and the all-of-the-above aggregate do not.

Why each choice is right or wrong

A - Correct: Binary means at most two children per node.

B - Incorrect: An equal key violates the specified strict left-subtree ordering.

C - Correct: Distinct keys mean every stored key value is unique.

D - Incorrect: B is false, so not all preceding statements hold.

Primary note proof

Study-note section DSA18

Binary means at most two children. Full means each node has zero or two children. Complete means all earlier levels are filled and the last fills from left. Perfect means all leaves share a level and every internal node has two children. Balanced means height stays logarithmic under a stated rule; it does not require a perfect shape.

Study-note section DSA19

With distinct keys, every key in a node's left subtree is smaller and every key in its right subtree is larger. Immediate-child comparisons alone are insufficient: root 10, left child 5, and 5's right child 12 violate the root's restriction. Duplicates require a separate count or placement policy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Question sources

Computer Science model exam 2 · Question 52

Question 85 | Q00712

The operation for adding an item to a stack is named to be___.

A. Push

B. Insert

C. Add

D. enqueue

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Push

The module names the stack addition operation Push. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Correct: Push is the defined stack insertion term. See assigned PDF 6.

B - Incorrect: Insert is a generic verb, not the given operation name. See assigned PDF 6.

C - Incorrect: Add is descriptive, not the stack-specific name. See assigned PDF 6.

D - Incorrect: Enqueue is the queue insertion name. See assigned PDF 6.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science model exam 2 · Question 53

Question 86 | Q00713

_____Which operation in stack could result underflow?

A. Push();

B. Is_empty();

C. Pop();

D. B and C

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Pop();

Only an attempted pop removes an element. When top==0 no element exists, so this removal attempt underflows; checking emptiness does not consume an element. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Incorrect: Push can encounter a full stack, not empty-removal underflow. See assigned PDF 6, 6.

B - Incorrect: Is_empty only checks state. See assigned PDF 6, 6.

C - Correct: Pop attempts removal and can find the stack empty. See assigned PDF 6, 6.

D - Incorrect: B does not underflow, so B and C is false. See assigned PDF 6, 6.

Primary note proof

Study-note section DSA11

> Overflow is an attempted insertion into full storage; underflow is an attempted removal from empty storage. Merely checking isEmpty does not underflow.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 24-42

> I. Array implementation (static memory).
> Push(n): Adding item at top
> top is the current top of stack and n is its
> maximum size
> Begin
> if (top == n) then stack is full;
> else
> top := top+1;
> stack[top] := item;
> end:

```
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;
```

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science model exam 2 · Question 54

Question 87 | Q00714

During ordinary lookup in a binary search tree with distinct keys and strict left-smaller/right-larger ordering, which rule correctly chooses the next subtree or recognizes a match?

A. If the searched key is less than the root, go to left.

B. If the searched key is larger than the root, go to left.

C. If the searched key is equal to the root, go to left.

D. None.

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. If the searched key is less than the root, go to left.

The BST invariant puts every key in the current node’s left subtree below that node’s key and every right-subtree key above it. A target smaller than the current key must therefore be sought on the left. A larger target belongs on the right, while equality finishes the lookup at the current node. This directional rule relies on BST ordering, not merely on having a binary shape.

Why each choice is right or wrong

A - Correct: A smaller target follows the left subtree under the BST invariant.

B - Incorrect: A larger target follows right, not left.

C - Incorrect: Equality is a successful match rather than a reason to continue left.

D - Incorrect: A supplies a valid search rule.

Primary note proof

Study-note section DSA19

> With distinct keys, every key in a node's left subtree is smaller and every key in its right subtree is larger. Immediate-child comparisons alone are insufficient: root 10, left child 5, and 5's right child 12 violate the root's restriction. Duplicates require a separate count or placement policy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

> • Binary tree: a tree in which each node has at most two children called left child and right child.
> • Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
> satisfies the following.
> o Every node has a key and no two elements have the same key.
> o The keys in the right subtree are larger than the keys in the root.
> o The keys in the left subtree are smaller than the keys in the root.
> o The left and the right subtrees are also binary search trees.

Question sources

Computer Science model exam 2 · Question 55

Question 88 | Q00715

Which key word is used to de-allocate the space taken by a node?

A. delete

B. alloc()

C. del()

D. A and B

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: delete

The deletion section explicitly uses delete temp to release node storage. Evidence: assigned PDF pages 5.

Why each choice is right or wrong

A - Correct: delete releases the pointed-to memory. See assigned PDF 5.

B - Incorrect: alloc is not the given C++ deallocation keyword. See assigned PDF 5.

C - Incorrect: del is not the module keyword. See assigned PDF 5.

D - Incorrect: B is not the prescribed deallocation instruction. See assigned PDF 5.

Primary note proof

Study-note section DSA03

> A C struct groups named fields, including arrays and pointers. A singly linked node needs payload plus a same-type link. C fields cannot have initializers inside the type declaration; modern C++ permits default member initializers. In C++, match new with delete and new[] with delete[]. Repair links before freeing a node; never dereference it afterwards.

Study-note section DSA09

> Prepend with new.next=head; head=new. To insert after a known predecessor p, set new.next=p.next before p.next=new. Appending with only head requires a tail traversal; maintaining tail avoids it. Delete the head by saving old=head, advancing head=head.next and releasing old in C++.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 5 | lines 35-41

> IV. Deleting a node from the list
> When a node is deleted, the space that it took up should be reclaimed. Otherwise the computer will eventually run out of memory space. This is done with the delete instruction:
> delete temp; // Release the memory pointed to by temp
> However, we can't just delete the nodes willy-nilly as it would break the chain. We need to reassign the pointers and then delete the node at the last moment. We can delete nodes at the start, middle or end of the list. Here let’s see the steps to delete at the end.

Question sources

Computer Science model exam 2 · Question 56

Question 89 | Q00716

Which one is easier to perform?

A. Deleting a leaf node

B. Deleting internal node

C. deleting root node

D. A and B

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: Deleting a leaf node

The module calls leaf deletion simple and straightforward: detach its parent link. Internal and root deletion can need child promotion or copying. Evidence: assigned PDF pages 10.

Why each choice is right or wrong

A - Correct: A leaf has no descendant subtree to reconnect. See assigned PDF 10.

B - Incorrect: An internal node can require preserving one or two child subtrees. See assigned PDF 10.

C - Incorrect: Root deletion may require choosing and reconnecting a replacement. See assigned PDF 10.

D - Incorrect: Internal deletion is not always as simple as leaf deletion. See assigned PDF 10.

Primary note proof

Study-note section DSA21

> A leaf can be detached by clearing its parent link. For one child, make the parent bypass the node and point to the child. For two children, copy the inorder predecessor (left-subtree maximum) or successor (right-subtree minimum), then delete that replacement from its original location. That second deletion has at most one child.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 10 | lines 5-19

> V. Deletion
> To delete a node (whose Num value is N) from binary search tree (whose root node is pointed by RootNodePtr), four cases should be considered. When a node is deleted the definition of binary search tree should be preserved.
> Case 1: Deleting a leaf node (a node having no child): it is simple and straightforward operation. Delete the leaf node and make the pointer of the deleted node parent NULL (right or left child pointer based on whether the deleted child is left child or right child)
> Case 2: Deleting a node having only one child

Case 3: Deleting a node having two children
Case 4: Deleting the root node
Note: For case 2, 3 and 4 there are two approaches
Approach 1: Deletion by copying
• Copy the node containing the largest element in the left (or the smallest element in the right) to the node containing the element to be deleted
• Delete the Copied Node

Question sources

Computer Science model exam 2 · Question 57

Question 90 | Q00717

Algorithms must terminate after a finite number of steps; What is this characteristics called__

A. Unambiguous

B. Feasibility

C. Finiteness

D. Independent

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Finiteness

The algorithm criteria expressly name termination after a finite number of steps Finiteness. Evidence: assigned PDF pages 2.

Why each choice is right or wrong

A - Incorrect: Unambiguous describes definiteness of each instruction. See assigned PDF 2.

B - Incorrect: Feasibility is not the term attached to termination here. See assigned PDF 2.

C - Correct: Finiteness is the named finite-termination criterion. See assigned PDF 2.

D - Incorrect: Independent does not name finite termination. See assigned PDF 2.

Primary note proof

Study-note section DSA04

Finiteness means termination after finitely many steps for every valid input. Definiteness means each instruction has one interpretation; effectiveness means each step is executable. Correctness means meeting the stated result for every valid input. A finite program can still return the wrong answer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 26-32

> Algorithm: An algorithm is a finite set of instructions which, if followed, accomplish a particular task. In addition every algorithm must satisfy the following criteria:
> • Input:there are zero or more quantities which are externally supplied;
> • Output: at least one quantity is produced;
> • Definiteness: each instruction must be clear and unambiguous;
> • Finiteness: if we trace out the instructions of an algorithm, then for all cases the algorithm will terminate after a finite number of steps;

Question sources

Computer Science model exam 2 · Question 58

Question 91 | Q00902

Which listed structures are linear data structures in the course classification? Select all that apply.

A. Tree

B. Array

C. Queue

D. Linked list

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B, C, D: B. Array; C. Queue; D. Linked list

Arrays, queues and linked lists arrange their items as linear sequences in this classification. A tree instead models hierarchical branching and is classified as nonlinear. Thus three offered structures meet the question’s category. The stray extraction characters before Queue are removed so the option states the structure’s name clearly, and the select-all instruction accounts for every valid member of the category.

Why each choice is right or wrong

A - Incorrect: A tree is hierarchical rather than linear.

B - Correct: An array is a linear structure.

C - Correct: A queue is a linear structure.

D - Correct: A linked list is a linear structure.

Primary note proof

Study-note section DSA01

An abstract data type (ADT) defines values and operations; representation supplies implementation. Arrays or linked nodes can implement the same LIFO stack. Abstraction keeps relevant details: patient urgency and arrival order matter for service; hair colour does not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 15-18

a. Linear data structure: data items stores in sequence
Example: Array, stack and Queue,
b. Non-Linear data structures : in which there is order of data items
Example: Trees and Graphs

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Question sources

Computer Science Exit Exam · 2018 · Question 15

Question 92 | Q00931

Which abbreviations describe stack service order? Select all that apply.

A. LIFO

B. FIFO

C. FILO

D. FCFS

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A, C: A. LIFO; C. FILO

A stack removes the latest inserted item first, described as last in, first out or LIFO. The equivalent view says that the first inserted item leaves last, described as first in, last out or FILO. Both abbreviations name the same policy. FIFO and FCFS instead serve earlier arrivals before later ones and describe the ordinary queue discipline rather than the stack discipline.

Why each choice is right or wrong

A - Correct: LIFO means last in, first out.

B - Incorrect: FIFO is the ordinary queue discipline.

C - Correct: FILO is equivalent to LIFO.

D - Incorrect: FCFS serves arrivals in queue order.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA14

> This is FIFO or first-come, first-served.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science Exit Exam · 2018 · Question 44

Question 93 | Q00947

In singly linked list, the node contains

A. Data and index

B. Data and one pointer (the next node)

C. Only data

D. Data and two pointers

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: Data and one pointer (the next node)

The singly linked node contains a data payload and a single pointer to the next node. Evidence: assigned PDF pages 4.

Why each choice is right or wrong

A - Incorrect: An index is not the described next-pointer field. See assigned PDF 4.

B - Correct: Both required components are present. See assigned PDF 4.

C - Incorrect: Payload alone cannot connect the chain. See assigned PDF 4.

D - Incorrect: Two pointers describe a different representation, not the supplied single-link node. See assigned PDF 4.

Primary note proof

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various

operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Question sources

Computer Science Exit Exam · 2018 · Question 60

Question 94 | Q00951

Which data structure follows LIFO principle?

A. Graphs

B. Queues

C. Stacks

D. Trees

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Stacks

Stack removes its most recently inserted top, exactly LIFO. Evidence: assigned PDF pages 6, 7, 2.

Why each choice is right or wrong

A - Incorrect: A graph structure does not prescribe a last-in deletion order. See assigned PDF 6, 7, 2.

B - Incorrect: Queue is FIFO. See assigned PDF 6, 7, 2.

C - Correct: Stack is expressly LIFO. See assigned PDF 6, 7, 2.

D - Incorrect: A tree hierarchy does not impose stack order. See assigned PDF 6, 7, 2.

Primary note proof

Study-note section DSA11

Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

“A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some

useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 7 | lines 10-16

"A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT". Queues are sometimes referred to as First in First out (FIFO) lists.
The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 2 | lines 15-18

a. Linear data structure: data items stores in sequence
Example: Array, stack and Queue,
b. Non-Linear data structures : in which there is order of data items
Example: Trees and Graphs

Question sources

Computer Science Exit Exam · 2018 · Question 64

Question 95 | Q00955

The time complexity for binary search is

A. O(n*log(n)2)

B. O(n2)

C. ¢.
O(n)

D. O(logn)

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer D: O(logn)

The source states search time proportional to log2 n; repeated midpoint selection halves the remaining ordered interval. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: n times a logarithmic factor is not the tight single-search bound. See assigned PDF 3.

B - Incorrect: Quadratic does not describe a halving path. See assigned PDF 3.

C - Incorrect: Linear is sequential-search growth. See assigned PDF 3.

D - Correct: Logarithmic matches the binary-search statement. See assigned PDF 3.

Primary note proof

Study-note section DSA29

Worst probes are floor($\log_2 n$)+1 for n>0: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Study-note section DSA06

For eventually nonnegative functions, f(n)=O(g(n)) means f(n)≤c g(n) for some positive c beyond a threshold k. Omega reverses the inequality; Theta requires both. Little-o means f/g tends to zero; little-omega means it grows without bound. These relations can describe best-case, average-case, worst-case or space functions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Question sources

Computer Science Exit Exam · 2018 · Question 68

Question 96 | Q00974

Which sorting algorithm compares each pair of adjacent elements from the beginning of an array and, if they are in reversed order, swaps them; if at least one swaps has been done, repeat step 1.

A. Insertion sort

B. Bubble sort

C. .

Selection sort

D. Shell sort

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer B: Bubble sort

The stem describes complete adjacent-pair passes and repeats if swaps occur. That is bubble sort with a no-swap stopping test. Evidence: assigned PDF pages 4, 3.

Why each choice is right or wrong

A - Incorrect: Insertion places a new item in the sorted prefix. See assigned PDF 4, 3, 3.

B - Correct: Bubble compares and exchanges adjacent out-of-order entries across passes. See assigned PDF 4, 3, 3.

C - Incorrect: Selection searches for a minimum rather than exchanging every reversed adjacent pair. See assigned PDF 4, 3, 3.

D - Incorrect: Shell first uses gap-separated subsequences, unlike these adjacent passes. See assigned PDF 4, 3, 3.

Primary note proof

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right. The largest remaining value moves to the final unsorted position after one left-to-right pass. Shrink the next pass by one position. Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

> C. Bubble Sort
> The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
> the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.

• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Question sources

Computer Science Exit Exam · 2018 · Question 87

Question 97 | Q00978

Among the following one is NOT the application of queue in the real world.

A. It is used in operating systems for handling interrupts.

B. It is used to maintain the play list in media players in order

C. .
It is used in Depth First search

D. It is widely used as waiting lists for a single shared resource

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer C: Depth-first search uses a stack rather than the ordinary FIFO queue.

DSA14 gives FIFO waiting, asynchronous buffers, and play sequences as queue uses. DSA42 explicitly contrasts DFS stack or recursion with a queue, which would change the discipline. Pending interrupts can be buffered for processing; this does not mean every interrupt policy must be FIFO. Standard depth-first search is the listed application that requires latest-pending-work service instead.

Why each choice is right or wrong

A - Incorrect: Pending interrupt work can be queued for handling; the claim does not require every interrupt policy to be FIFO.

B - Incorrect: A queue can preserve a play sequence in arrival order, as noted in DSA14.

C - Correct: Standard DFS uses a stack or recursion to finish a branch before returning, not a FIFO frontier.

D - Incorrect: First-come shared-resource waiting is a direct FIFO queue application described in DSA14.

Primary note proof

Study-note section DSA-S02

> An interrupt service routine may need to record work quickly and defer its completion. Operating systems can queue that deferred work for later processing; Windows deferred procedure calls are one concrete example. This is a queue application, but it does not imply that every interrupt system uses a single FIFO service policy.

Study-note section DSA14

> FIFO suits equal-priority printer jobs, shared-resource waiting, asynchronous buffers and level-order exploration. A play sequence can also use a queue. Distinguish airline seat records from waiting requests; an application name alone does not force one representation.

Study-note section DSA42

> Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours. An explicit stack can store pending work instead. Both implement latest-pending-work first; a queue would change that discipline. Marking prevents infinite repetition around graph cycles.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 41-48

> Application of Queue
> Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
> • Print server- maintains a queue of print jobs
> • Disk Driver- maintains a queue of disk input/output requests
> • Task scheduler in multiprocessing system- maintains priority queues of processes
> • Telephone calls in a busy environment -maintains a queue of telephone calls

• Simulation of waiting line- models real life queues (e.g. supermarkets checkout, customer service line (Bank, custom, airport etc))

Data Structure and Algorithms module | PDF page 7 | lines 2-8

• Reversing Data and Reverse a List
• Convert Decimal to Binary
• Parsing - It is a logic that breaks into independent pieces for further processing
• Backtracking
Examples:
1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Question sources

Computer Science Exit Exam · 2018 · Question 91

Question 98 | Q00991

An airline has a fixed finite seat layout, and each seat number maps directly to a dense integer position. Which listed representation gives constant-time lookup of that seat's reservation record under the usual RAM model?

A. Array

B. Linked List

C. Queue

D. Stack

Hint

Fix the input case and cost model, then count dominant work or simultaneously live storage.

Answer A: A. Array

The explicit seat-number mapping turns each lookup into an indexed-position query over a fixed finite layout. An array can compute the corresponding element address directly, giving constant-time access in the note's RAM model. An ordinary linked list must traverse to a position, while queues and stacks expose service endpoints rather than arbitrary seat lookup. This models seat records themselves, rather than a queue of waiting reservation requests.

Why each choice is right or wrong

A - Correct: Direct seat-position indexing gives constant-time array lookup.

B - Incorrect: An ordinary linked list needs positional traversal.

C - Incorrect: A queue exposes front/rear service, not direct seat lookup.

D - Incorrect: A stack exposes its top rather than arbitrary seat positions.

Primary note proof

Study-note section DSA02

For equal-size elements, address(A[i]) = base + i × element_size under zero-based indexing. This arithmetic gives constant-time positional access in the usual RAM cost model. A row-major rectangular matrix uses base + (row × columns + column) × element_size.

Study-note section DSA08

A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 41-48

Application of Queue
Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
• Print server- maintains a queue of print jobs
• Disk Driver- maintains a queue of disk input/output requests
• Task scheduler in multiprocessing system- maintains priority queues of processes
• Telephone calls in a busy environment -maintains a queue of telephone calls
• Simulation of waiting line- models real life queues (e.g. supermarkets checkout, customer service line (Bank, custom, airport etc))

Question sources

National Exit Exam practice questions · Question 5

Computer Science Exit Exam practice collection · Question 129

Question 99 | Q01005

Which notation writes binary arithmetic operators between their operands and uses precedence and parentheses, as in ordinary written arithmetic?

A. Infix notation

B. Prefix notation

C. Postfix notation

D. None of the above

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. Infix notation

Infix notation places each binary operator between its two operands, using precedence rules and parentheses to establish grouping. Prefix puts the operator before its operands, and postfix puts it after them. The criterion asks for that objective familiar arithmetic layout rather than a universal claim about personal comfort. Since A names the between-operands form, the none-of-the-above alternative does not apply.

Why each choice is right or wrong

A - Correct: Infix places the operator between operands.

B - Incorrect: Prefix puts the operator before its operands.

C - Incorrect: Postfix puts the operator after its operands.

D - Incorrect: Infix satisfies the specified layout.

Primary note proof

Study-note section DSA13

> Infix places an operator between operands and uses precedence and parentheses. Prefix places it first; postfix places it last. Their fixed operator arity determines grouping. To convert infix to postfix, output operands immediately and hold operators on a stack. Pop higher-precedence operators before pushing a new operator; also pop equal precedence for left-associative operators. Parentheses temporarily override this order.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

> • Reversing Data and Reverse a List
> • Convert Decimal to Binary
> • Parsing - It is a logic that breaks into independent pieces for further processing
> • Backtracking
> Examples:
> 1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
> 2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Question sources

National Exit Exam practice questions · Question 19

Computer Science Exit Exam practice collection · Question 130

Question 100 | Q01181

Which option is outside the three standard named depth-first visitation orders for binary trees in this course? This asks about those standard order names, not whether other traversal algorithms can exist.

A. Randomized traversal

B. Preorder traversal

C. Postorder traversal

D. Inorder traversal

Hint

Compare the required access order and representation constraints with the defining operations of each option.

Answer A: A. Randomized traversal

The three standard named orders are preorder, inorder and postorder, determined respectively by visiting the node before, between or after its two child subtrees. Randomized visitation can be implemented, but it is not one of these three conventional names. Thus Randomized traversal is the correct exclusion under the explicit classification. The question no longer claims that such a traversal is impossible or never used.

Why each choice is right or wrong

A - Correct: Randomized visitation is possible but is not one of the three standard named orders being classified.

B - Incorrect: Preorder is one of the standard orders.

C - Incorrect: Postorder is one of the standard orders.

D - Incorrect: Inorder is one of the standard orders.

Primary note proof

Study-note section DSA20

> Preorder visits node,left,right; inorder left,node,right; postorder left,right,node. Apply recursively; null children do nothing. Inorder returns sorted keys only for a BST. Preorder supports copying, postorder supports deleting children before parents, and level order uses a FIFO queue.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 40-44

> III. Tree Traversing:
> Binary search tree can be traversed in three ways.
> a. Pre order traversal - traversing binary tree in the order of parent, left and right.

b. Inorder traversal - traversing binary tree in the order of left, parent and right.
c. Postorder traversal - traversing binary tree in the order of left, right and parent.

Question sources

Computer Science model exam 2 · Question 2

Question 101 | Q01182

Which of the following principles does Queue use?

A. LIFO principle

B. FIFO principle

C. Linear tree

D. Ordered array

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: FIFO principle

Insertion at rear and deletion at front preserve arrival order, which the source calls FIFO. Evidence: assigned PDF pages 7.

Why each choice is right or wrong

A - Incorrect: LIFO is the stack policy. See assigned PDF 7.

B - Correct: FIFO is explicitly named for queues. See assigned PDF 7.

C - Incorrect: Linear tree is not an ordering policy. See assigned PDF 7.

D - Incorrect: Ordered array describes neither the queue service rule nor all its representations. See assigned PDF 7.

Primary note proof

Study-note section DSA14

An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

“A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.

The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Computer Science model exam 2 · Question 3

Question 102 | Q01198

In this course's standard ADT vocabulary, which structure calls insertion at its top push and removal from that same top pop?

A. Queue

B. Stack

C. Both

D. None

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: B. Stack

The stack contract inserts and removes at one top. Its standard operations are push for insertion and pop for removal, with peek observing the top without removing it. The ordinary queue contract instead enqueues at the rear and dequeues at the front. The question fixes this conceptual vocabulary rather than comparing method names chosen by particular programming libraries, making Stack the definite selection.

Why each choice is right or wrong

A - Incorrect: The standard queue vocabulary is enqueue/dequeue at different endpoints.

B - Correct: Push and pop name the standard stack-top operations.

C - Incorrect: Both would incorrectly include the queue under the specified vocabulary.

D - Incorrect: Stack is present and matches the stated operations.

Primary note proof

Study-note section DSA11

Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA14

An ordinary queue enqueues at rear, dequeues at front, and peeks at front.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

"A stack is an ordered list in which all insertions and deletions are made at one end, called the top". Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 7 | lines 10-16

"A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT". Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Computer Science model exam 2 · Question 19

Question 103 | Q01204

Which task requires FIFO service rather than the LIFO service of a stack used as the sole pending-item ADT? In D, asynchronous transfer must deliver buffered items in arrival order.

A. String reversal

B. Recursion

C. Backtracking

D. Asynchronous buffered transfer with mandatory arrival-order delivery

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: D. Asynchronous buffered transfer with mandatory arrival-order delivery

Stacks support reversal by popping items in reverse insertion order. They also manage nested recursive calls and backtracking through the most recent pending context or choice. The stated asynchronous-transfer contract instead requires older buffered items to be delivered before newer arrivals. That is FIFO service, so an ordinary queue supplies the needed ordering. A single stack's LIFO removal rule does not meet this explicitly ordered buffering contract.

Why each choice is right or wrong

A - Incorrect: Reversal uses the stack's reverse insertion order.

B - Incorrect: Nested recursive calls return most-recent-first.

C - Incorrect: Backtracking resumes the latest pending choice.

D - Correct: Arrival-order delivery requires FIFO rather than a single stack's LIFO.

Primary note proof

Study-note section DSA12

> Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Study-note section DSA14

> FIFO suits equal-priority printer jobs, shared-resource waiting, asynchronous buffers and level-order exploration. A play sequence can also use a queue. Distinguish airline seat records from waiting requests; an application name alone does not force one representation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

> • Reversing Data and Reverse a List
> • Convert Decimal to Binary
> • Parsing - It is a logic that breaks into independent pieces for further processing
> • Backtracking
> Examples:
> 1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
> 2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Data Structure and Algorithms module | PDF page 6 | lines 46-52

> Applications of Stack
> • It is very useful to evaluate arithmetic expressions. (Postfix Expressions)
> • Infix to Postfix Transformation
> • It is useful during the execution of recursive programs
> • A Stack is useful for designing the compiler in operating system to store local

variables inside a function block.
• A stack (memory stack) can be used in function calls including recursion.

Question sources

Computer Science model exam 2 · Question 25

Question 104 | Q01228

For an unordered symbol table with no range queries, assume suitable hashing and bounded load factor. Which listed ordinary representation offers expected constant-time lookup, compared with linear list search and logarithmic balanced-tree search?

A. Self-organizing list

B. Hash table

C. Balanced search tree

D. Linear list scan

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: B. Hash table

Symbol tables can use hashing for expected constant lookup when hashing spreads keys suitably and load remains bounded. Balanced search trees supply logarithmic lookup and ordering, while list search can require a linear scan. The comparison concerns the expected lookup growth under those assumptions, not an unconditional worst-case minimum. Collision-heavy hash inputs can still take linear time outside that expected analysis.

Why each choice is right or wrong

A - Incorrect: Ordinary self-organizing list search does not guarantee the stated general expected bound.

B - Correct: Suitable hashing with bounded load provides expected constant lookup.

C - Incorrect: A balanced search tree has logarithmic lookup.

D - Incorrect: A linear list scan takes linear worst-case lookup.

Primary note proof

Study-note section DSA27

A dictionary or compiler symbol table can use hashing for expected constant lookup, or a balanced tree for logarithmic lookup with ordering. A bounded cache also needs eviction rules: FIFO uses a queue; LRU can combine hash lookup with a doubly linked recency list. Move a hit to the recent end and evict the least-recent end when full. No storage structure is universally best without operation and policy requirements.

Study-note section DSA26

Separate chaining stores colliding entries in a bucket list. Load alpha=n/m is average chain length; expected operations are O(1+alpha) under suitable distribution assumptions, with O(n) worst cases. Open addressing keeps entries inside the array and probes until a key or an available slot is found.

Question sources

Computer Science model exam 2 · Question 49

Question 105 | Q01241

In the linked list implementation of a queue, where will the new element be inserted?

A. At the middle position of the linked list.

B. At the head position of the linked list.

C. At the tail position of the linked list.

D. None of the above.

Hint

Trace the initial state and update only the fields affected by each permitted operation.

Answer C: At the tail position of the linked list.

The linked queue implementation expressly inserts a node at the end and removes the first node. End means tail. Evidence: assigned PDF pages 8.

Why each choice is right or wrong

A - Incorrect: Middle insertion would break the specified FIFO insertion rule. See assigned PDF 8.

B - Incorrect: Head insertion is the removal end in this representation. See assigned PDF 8.

C - Correct: New nodes are inserted at the linked-list end/tail. See assigned PDF 8.

D - Incorrect: C matches the implementation. See assigned PDF 8.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 35-37

> III. Linked list implementation of enqueue and dequeue operations
> Enqueue- is inserting a node at the end of a linked list
> Dequeue- is deleting the first node in the list

Question sources

Computer Science model exam 2 · Question 62

Question 106 | Q01263

What is the maximum number of children that a node can have in a binary tree?

A. 3

B. 1

C. 4

D. 2

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: 2

A binary tree has at most two children per node, named left and right. Maximum is therefore two. Evidence: assigned PDF pages 9.

Why each choice is right or wrong

A - Incorrect: Three exceeds the binary limit. See assigned PDF 9.

B - Incorrect: One is permitted but is not the maximum. See assigned PDF 9.

C - Incorrect: Four exceeds the binary limit. See assigned PDF 9.

D - Correct: Two equals the specified maximum. See assigned PDF 9.

Primary note proof

Study-note section DSA18

> Binary means at most two children. Full means each node has zero or two children. Complete means all earlier levels are filled and the last fills from left. Perfect means all leaves share a level and every internal node has two children. Balanced means height stays logarithmic under a stated rule; it does not require a perfect shape.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 14-19

> • Ancestors of a node: parent, grandparent, grand-grandparent, etc of a node.
> • Descendants of a node: children, grandchildren, grand-grandchildren etc of a node.
> • Depth of a node: number of ancestors or length of the path from the root to the node.
> • Height of a tree: depth of the deepest node.
> • Subtree: a tree consisting of a node and its descendants.
> • Binary tree: a tree in which each node has at most two children called left child and right child.

Question sources

Computer Science model exam 2 · Question 84

Question 107 | Q01442

DAG is an abbreviation of?

A. Detecting Acyclic Graph

B. Data Acyclic Graph

C. Dynamic Acyclic Graph

D. Directed Acyclic Graph

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: Directed Acyclic Graph

DSA40 directly expands DAG to directed acyclic graph. Directed specifies that each edge has an orientation, and acyclic excludes directed cycles. Detecting, Data, and Dynamic do not supply the defined first word. This is a vocabulary question and does not depend on graph representation or a particular traversal algorithm.

Why each choice is right or wrong

A - Incorrect: Detecting is not the first word in the defined expansion of DAG.

B - Incorrect: Data is not the first word in the defined graph-theoretic abbreviation DAG.

C - Incorrect: Dynamic is not the first word of DAG as defined in DSA40.

D - Correct: Directed Acyclic Graph is the exact expansion explicitly given in the complete note.

Primary note proof

Study-note section DSA40

> A graph has vertices V and edges E. Directed edges distinguish u→v from v→u; undirected edges represent two-way relationships. Following accounts is directed; a mutual friendship relation is undirected. Weighted edges carry costs; unweighted shortest-path questions count edges. DAG expands to directed acyclic graph. Trees are connected acyclic graphs, although graph data structures may also contain cycles.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 15-18

> a. Linear data structure: data items stores in sequence
> Example: Array, stack and Queue,
> b. Non-Linear data structures : in which there is order of data items
> Example: Trees and Graphs

Question sources

Compiler Design review questions · Question 50

Question 108 | Q01497

A binary search tree was built by inserting the following numbers in the given order: 40, 20, 50, 10, 30, 60. Which of the following statements are true?

A. 30 is the left child of 40

B. the tree is a heap

C. height of the tree is 3

D. 50 is the right child of 40

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: 50 is the right child of 40

Insertions give root 40; its children 20 and 50; 20 has children 10 and 30; 50 has right child 60. The deepest depth is two edges. A max heap is impossible because child 50 exceeds root 40. Evidence: assigned PDF pages 9, 11.

Why each choice is right or wrong

A - Incorrect: 30 is the right child of 20, not the left child of 40. See assigned PDF 9, 9, 9, 11.

B - Incorrect: 50 exceeds its parent 40, violating the module max-heap rule. See assigned PDF 9, 9, 9, 11.

C - Incorrect: Height is depth two, not the three-level count. See assigned PDF 9, 9, 9, 11.

D - Correct: 50 is larger than 40 and becomes its right child. See assigned PDF 9, 9, 9, 11.

Primary note proof

Study-note section DSA19

> With distinct keys, every key in a node’s left subtree is smaller and every key in its right subtree is larger. Immediate-child comparisons alone are insufficient: root 10, left child 5, and 5’s right child 12 violate the root’s restriction. Duplicates require a separate count or placement policy.

Study-note section DSA18

> A rooted tree has one parentless root and one parent per other node. It is connected and acyclic, has n-1 edges, and one path between any pair. Depth counts root-to-node edges; height is maximum depth. A single-node tree therefore has height zero. Leaves have no children.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

> • Binary tree: a tree in which each node has at most two children called left child and right child.
> • Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
> satisfies the following.

o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Data Structure and Algorithms module | PDF page 9 | lines 33-39

II. Insertion
When a node is inserted the definition of binary search tree should be preserved.
Case 1: There is no data in the tree (i.e. RootNodePtr is NULL)
• The node pointed by InsNodePtr should be made the root node.
Case 2: There is data
• Search the appropriate position.
• Insert the node in that position.

Data Structure and Algorithms module | PDF page 9 | lines 14-19

• Ancestors of a node: parent, grandparent, grand-grandparent, etc of a node.
• Descendants of a node: children, grandchildren, grand-grandchildren etc of a node.
• Depth of a node: number of ancestors or length of the path from the root to the node.
• Height of a tree: depth of the deepest node.
• Subtree: a tree consisting of a node and its descendants.
• Binary tree: a tree in which each node has at most two children called left child and right child.

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Question sources

Mock exam 1 · Question 13

Question 109 | Q01498

Certain operations on lists can be performed faster when the list is sorted than not. Which of the following could be an example of such an operation?

A. Searching for an element of a list

B. Computing the variance of a list

C. Computing the average of a list

D. All can be performed faster when the list is sorted.

E. Counting for the number of odd numbers in a list

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A: Searching for an element of a list

Sortedness permits logarithmic binary search instead of a linear scan. Sorting alone does not remove the need to inspect all values for mean, variance or arbitrary parity counts. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Correct: Binary searching uses sorted order to discard half the list at each step. See assigned PDF 3, 3.

B - Incorrect: Variance still depends on all values, without extra aggregates. See assigned PDF 3, 3.

C - Incorrect: Average still depends on all values, without a stored sum. See assigned PDF 3, 3.

D - Incorrect: The statistical computations are not accelerated solely by ordering. See assigned PDF 3, 3.

E - Incorrect: Odd and even values are not a monotone split in numeric sorted order. See assigned PDF 3, 3.

Primary note proof

Study-note section DSA28

> For a single unsorted query, sorting first is rarely justified. Small lists may favour scanning because setup matters. Sorting does not avoid examining all values for mean, variance or arbitrary parity counts.

Study-note section DSA29

> On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Mock exam 1 · Question 14

Question 110 | Q01499

For the values 5 3 8 9 1 7 0 2 6 4, which offered arrays satisfy the final-pivot partition condition for pivot 5: all smaller values precede it and all larger values follow it? Do not require a particular partition algorithm or internal ordering. Select all that apply.

A. 3 1 0 2 4 5 8 9 7 6

B. None of the choices are correct

C. 4 2 3 0 1 5 7 9 6 8

D. 0 3 4 2 1 5 7 9 6 8

E. 3 1 0 2 4 5 8 9 6 7

F. 5 3 4 2 1 0 7 9 6 8

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A, C, D, E: A. 3 1 0 2 4 5 8 9 7 6; C. 4 2 3 0 1 5 7 9 6 8; D. 0 3 4 2 1 5 7 9 6 8; E. 3 1 0 2 4 5 8 9 6 7

There are five values below 5 and four values above it, so a final-position pivot must occupy index five in zero-based order. A, C, D and E each place precisely the smaller group before that pivot and the larger group after it. Their internal subgroup orders need not agree because only the partition invariant is requested. F leaves smaller values after an initial pivot and fails the condition.

Why each choice is right or wrong

A - Correct: All five smaller values precede 5 and all larger ones follow.

B - Incorrect: Several offered arrays satisfy the condition.

C - Correct: This array satisfies the smaller/pivot/larger grouping.

D - Correct: This array also satisfies the grouping despite different internal order.

E - Correct: The right-side permutation does not violate the grouping.

F - Incorrect: Smaller values after the leading 5 violate its final-position condition.

Primary note proof

Study-note section DSA36

> In the ten-distinct-value example, a final pivot 5 has five smaller values before it and four larger after it. Internal orders depend on the partition algorithm; several candidate arrays can satisfy the invariant. State the algorithm before predicting one exact array. Quicksort normally rearranges in the same array but recursive storage can be linear in the worst case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

> ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
> Algorithm:
> 1. Choose a pivot value (mostly the first element is taken as the pivot value)
> 2. Position the pivot element and partition the list so that:
> • the left part has items less than or equal to the pivot value
> • the right part has items greater than or equal to the pivot value
> 3. Recursively sort the left part
> 4. Recursively sort the right part

Question sources

Mock exam 1 · Question 15

Question 111 | Q01578

An owning dynamically allocated linked-list API must provide item search, insertion and removal. Removal unlinks and frees its node before returning; it returns any needed payload by value. Which additional function is unnecessary and would violate the contract if its result were dereferenced?

A. A function that returns a pointer to an already freed node for later dereferencing

B. A function that searches for an item from the list

C. A function that inserts an item in the list

D. A function that removes an item from the list

E. None of the choices is correct

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A: A. A function that returns a pointer to an already freed node for later dereferencing

The API explicitly requires search, insertion and removal, so those operations belong in its interface. Its removal contract frees the node before returning, leaving that node's storage unavailable for later dereference. An additional function promising a usable pointer to the freed node is therefore inappropriate. Returning a copied payload is different and remains allowed; the distinction concerns storage lifetime rather than whether deletion may report a value.

Why each choice is right or wrong

A - Correct: A usable pointer to a freed node violates the stated lifetime contract.

B - Incorrect: Item search is explicitly required by this API.

C - Incorrect: Item insertion is explicitly required by this API.

D - Incorrect: Item removal is explicitly required by this API.

E - Incorrect: A identifies an inappropriate extra function.

Primary note proof

Study-note section DSA03

> A C struct groups named fields, including arrays and pointers. A singly linked node needs payload plus a same-type link. C fields cannot have initializers inside the type declaration; modern C++ permits default member initializers. In C++, match new with delete and new[] with delete[]. Repair links before freeing a node; never dereference it afterwards.

Study-note section DSA09

> Prepend with new.next=head; head=new. To insert after a known predecessor p, set new.next=p.next before p.next=new. Appending with only head requires a tail traversal; maintaining tail avoids it. Delete the head by saving old=head, advancing head=head.next and releasing old in C++.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 5 | lines 35-41

IV. Deleting a node from the list
When a node is deleted, the space that it took up should be reclaimed. Otherwise the computer will eventually run out of memory space. This is done with the delete instruction:
delete temp; // Release the memory pointed to by temp
However, we can't just delete the nodes willy-nilly as it would break the chain. We need
to reassign the pointers and then delete the node at the last moment. We can delete nodes
at the start, middle or end of the list. Here let’s see the steps to delete at the end.

Question sources

Mock exam 3 · Question 10

Question 112 | Q01579

Consider a max-heap of n>=3 distinct keys, stored in ordinary level-order array form. Its binary tree is full: every internal node has two children. Which statement is false?

A. The root is the biggest element.

B. The second biggest element is necessarily found at depth 1 (root is at depth 0)

C. The search for the smallest element can be accomplished in O(log n) time (worst case)

D. The last element of the array is necessarily the right most child on the lowest level.

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: C is false: finding the minimum can require Theta(n) time.

The root contains the maximum key. With distinct keys, the second largest must be one of its children, since a deeper key has a larger ancestor above it. Completeness places the final occupied array slot at the rightmost occupied position of the lowest level. The minimum can occur at any leaf, however, so finding it may require a linear scan rather than logarithmic worst-case time. C is therefore false.

Why each choice is right or wrong

A - Incorrect: The max-heap order makes the root the largest key, so this is true rather than the requested false statement.

B - Incorrect: Distinct keys ensure the second largest is a root child, so this is true.

C - Correct: The minimum may require scanning linearly many leaves; the logarithmic worst-case claim is false.

D - Incorrect: Level-order storage of a complete heap ends at the rightmost occupied position on the lowest level.

Primary note proof

Study-note section DSA23

> A binary max-heap is complete with each parent at least its children; a min-heap reverses this order. Unlike BSTs, heaps do not order entire subtrees. The root is maximum. If all keys are distinct, the second largest is a root child. With duplicate maxima, the second largest distinct value may be deeper: [9,9,9,8,7,6,5] is a valid max-heap and that value is 8 at index 3. The minimum may be any leaf, requiring linear worst-case scanning.

Study-note section DSA18

> Complete means all earlier levels are filled and the last fills from left.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 23-37

> iii. Heap Sort
> Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
> each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
> Algorithm:
> 1. Construct a binary tree
> • The root node corresponds to Data[0].
> • If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.

2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Data Structure and Algorithms module | PDF page 9 | lines 14-19

- Ancestors of a node: parent, grandparent, grand-grandparent, etc of a node.
- Descendants of a node: children, grandchildren, grand-grandchildren etc of a node.
- Depth of a node: number of ancestors or length of the path from the root to the node.
- Height of a tree: depth of the deepest node.
- Subtree: a tree consisting of a node and its descendants.
- Binary tree: a tree in which each node has at most two children called left child and right child.

Question sources

Mock exam 3 · Question 11

Question 113 | Q01580

For ordinary top-down array mergesort on n=8 items with left-first ties and no merge-skipping, which description alone guarantees the maximum possible number of key comparisons across the merges?

A. When the list is initially sorted in the "reverse" order

B. When the list is already sorted in the "right" order

C. When the list contains all similar elements

D. When the list contains all distinct elements

E. None of the choices is correct

Hint

Compare the required operations, access order and representation metadata with each option.

Answer E: E. None of the choices is correct

Maximum merge comparisons depend on the interleaving of the sorted runs, rather than merely on a broad input label. Sorted or reverse-sorted input produces separated runs, and equal keys let left-first merging exhaust a run early. Distinct keys permit a worst interleaving but do not force one; already sorted distinct keys are a counterexample. None of A-D therefore guarantees the requested exact maximum.

Why each choice is right or wrong

A - Incorrect: Reverse order alone does not force worst interleavings.

B - Incorrect: Sorted runs can be merged with an early exhaustion of one side.

C - Incorrect: Left-first equal keys exhaust a run without maximum interleaving.

D - Incorrect: Distinct keys can already be sorted, so distinctness is insufficient.

E - Correct: No preceding description alone guarantees the exact maximum comparison count.

Primary note proof

Study-note section DSA37

> Split into halves, sort them, then merge by advancing the smaller current front. With left-first tie handling, equal keys retain order. Ordinary array merging needs a linear auxiliary buffer. There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA37

> Reverse order alone does not characterize maximum merge comparisons.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 38-44

> iv. Merge sort
> Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
> Algorithm:
> 1. Divide the array in to two halves.
> 2. Recursively sort the first n/2 items.
> 3. Recursively sort the last n/2 items.
> 4. Merge sorted items (using an auxiliary array).

Question sources

Mock exam 3 · Question 12

Question 114 | Q01618

If a sorted list contains 100 elements, what is the maximum number of comparisons to make to search for an element using binary searching algorithm?

A. None of the choices are correct

B. 100

C. 10

D. 7

E. 50

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: 7

With one midpoint-key comparison per visited node, the maximum probes are floor(log2 100)+1=7, since 64<=100<128. Equivalently seven binary levels can distinguish up to 127 keys. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: Seven is present as D. See assigned PDF 3.

B - Incorrect: 100 is a complete linear scan. See assigned PDF 3.

C - Incorrect: 10 exceeds the needed halving depth. See assigned PDF 3.

D - Correct: Seven midpoint probes suffice in the worst case. See assigned PDF 3.

E - Incorrect: 50 counts only the first halving, not the number of probes. See assigned PDF 3.

Primary note proof

Study-note section DSA29

> Worst probes are floor($\log_2 n$)+1 for n>0: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Question sources

Mock exam 2 · Question 7

Question 115 | Q01619

For ordinary representations without an extra positional index, which listed structure guarantees O(1) worst-case access to an arbitrary valid logical position as its length grows? Use fixed-size elements and unit-cost address arithmetic.

A. Singly linked list

B. Doubly linked list

C. Every listed linked structure guarantees constant-time access to an arbitrary logical position.

D. Circular linked list

E. Array

Hint

Compare the required operations, access order and representation metadata with each option.

Answer E: E. Array

An array computes an element address from its base address, the requested index and the fixed element size, giving constant-time access. Ordinary singly, doubly and circular linked lists instead follow links to reach a logical position; even starting from the nearer endpoint leaves linear worst-case traversal. The replacement aggregate claim is false because those linked structures lack a direct positional index. Array is therefore the unique matching choice.

Why each choice is right or wrong

A - Incorrect: A singly linked list may traverse linearly many successor links.

B - Incorrect: A doubly linked list still has linear worst-case access to middle positions.

C - Incorrect: The listed linked structures do not all have constant-time arbitrary positional lookup.

D - Incorrect: Circularity changes how a chain ends, not how a logical position is located.

E - Correct: An array computes the address directly with fixed-size element arithmetic.

Primary note proof

Study-note section DSA02

> For equal-size elements, address(A[i]) = base + i × element_size under zero-based indexing. This arithmetic gives constant-time positional access in the usual RAM cost model.

Study-note section DSA08

> An array-linked list still follows links unless separately position-indexed.

Study-note section DSA10

> Finding a position is still linear.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 5 | lines 26-34

III. Displaying the list of Nodes
After adding nodes to the list, we need to display the list of nodes on the screen. Here are the steps required to display all nodes item in the list:
1. Set a temporary pointer to point to the same thing as the start pointer.
2. If the pointer points to NULL, display the message "End of list" and stop.
3. Otherwise, display the details of the node pointed to by the start pointer.
4. Make the temporary pointer point to the same thing as the next pointer of the node it is currently indicating.
5. Jump back to step 2.

Data Structure and Algorithms module | PDF page 6 | lines 24-42

I. Array implementation (static memory).
Push(n): Adding item at top
top is the current top of stack and n is its
maximum size
Begin
if (top == n) then stack is full;
else
top := top+1;
stack[top] := item;
end:
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;

Question sources

Mock exam 2 · Question 8

Question 116 | Q01620

Which listed representations guarantee O(log n) worst-case arbitrary-key lookup under these implementations? Use a height-balanced BST for A, an ordinary singly linked list for C, an ordinary binary heap for D, and a sorted random-access array with binary search for E. Select all that apply.

A. Height-balanced BST

B. A and C

C. Singly linked list

D. Heap

E. Sorted random-access array with binary search

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A, E: A. Height-balanced BST; E. Sorted random-access array with binary search

A height-balanced BST has logarithmic height, so following its ordered search path takes logarithmic worst-case time. A sorted random-access array supports logarithmic binary-search probes. An ordinary singly linked list must traverse links, and a heap does not order entire subtrees by arbitrary search keys. Thus the combined A-and-C claim fails because C lacks the guarantee, while A and E independently meet it.

Why each choice is right or wrong

A - Correct: Balance bounds the BST search path logarithmically.

B - Incorrect: C lacks the logarithmic guarantee, so A and C is false.

C - Incorrect: Singly linked lookup may scan all nodes.

D - Incorrect: Heap order privileges the extreme but not arbitrary-key lookup.

E - Correct: Binary search on a sorted random-access array is logarithmic.

Primary note proof

Study-note section DSA19

> Search compares at each node and follows one child until equality or null. Insertion uses that same path to attach a new leaf. Operations cost O(height): logarithmic for balanced trees, linear for a chain. Best-case successful search is Theta(1), a root hit. An optimal BST minimizes expected search cost using access probabilities; constructing it is a different problem from searching it.

Study-note section DSA29

Worst probes are floor($\log_2 n$)+1 for n>0: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Study-note section DSA23

Unlike BSTs, heaps do not order entire subtrees.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates
mid-point and identify which have the target
belongs to since the list is sorted. It performs
this recursively till the item found or the list
is empty(target is not in the list
• The computational time for this algorithm is
proportional to log2 n.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Question sources

Mock exam 2 · Question 9

Question 117 | Q00426B

A queue is a ?

A. FIFO (First In First Out) list

B. LIFO (Last In First Out) list

C. Ordered array

D. Linear tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A: FIFO (First In First Out) list

The source expressly calls queues First in First out lists. Evidence: assigned PDF pages 7.

Why each choice is right or wrong

A - Correct: FIFO preserves arrival order. See assigned PDF 7.

B - Incorrect: LIFO names a stack. See assigned PDF 7.

C - Incorrect: An ordered array is one possible storage form, not the abstract queue rule. See assigned PDF 7.

D - Incorrect: Linear tree is not the source queue definition. See assigned PDF 7.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Data Structures & Algorithms practice questions · Question 56

Question 118 | Q00394

What is an internal sorting algorithm?

A. Algorithm that uses tape or disk during the sort

B. Algorithm that uses main memory during the sort

C. Algorithm that involves swapping

D. Algorithm that are considered 'in place'

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: Sorting whose working dataset fits in main memory.

DSA37 defines internal sorting by fitting the working dataset in memory. Option B expresses that main-memory setting. In contrast, external sorting handles disk-backed runs when data exceeds memory. Swapping is an operation used by some algorithms, and in-place describes auxiliary storage; neither determines whether a sort is internal.

Why each choice is right or wrong

A - Incorrect: Tape or disk working data characterizes external sorting rather than the main-memory internal setting.

B - Correct: Internal sorting keeps its working dataset in main memory, matching the definition in DSA37.

C - Incorrect: Swapping is an implementation operation and does not define the location of the working dataset.

D - Incorrect: In-place concerns auxiliary element storage; an internal sort may still use a separate memory buffer.

Primary note proof

Study-note section DSA37

> Top-down mergesort recurses; bottom-up merges runs of widths 1,2,4,... iteratively. In-place merging variants exist with different tradeoffs; linear merge names an operation, not a universally standardized sort variant. Internal sorting fits the working dataset in memory. External sorting builds sorted runs and merges buffered streams from disk, aiming to reduce I/O. Reverse order alone does not characterize maximum merge comparisons.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 33-37

> Algorithm analysis refers to the process of determining the amount of computing time and storage
> space required by different algorithms. It used to produce a function T (n) that describes the algorithm in terms of the operations performed in order to measure the complexityof the

algorithm.
Complexity Analysis is the systematic study of the cost of computation, measured either in time units or in operations performed, or in the amount of storage space required.

Question sources

Data Structures & Algorithms practice questions · Question 24

Question 119 | SM0008

What is the time complexity order of binary searching algorithm?

A. O(n^3)

B. O(n)

C. O(log n)

D. O(1)

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: O(log n)

The stated time is proportional to log2 n because each midpoint decision discards half the remaining ordered interval. Evidence: assigned PDF pages 3.

Why each choice is right or wrong

A - Incorrect: Cubic does not match repeated halving. See assigned PDF 3.

B - Incorrect: Linear is the source sequential-search order. See assigned PDF 3.

C - Correct: Logarithmic is the explicit binary-search order. See assigned PDF 3.

D - Incorrect: Constant time occurs for an immediate match, not a general search bound. See assigned PDF 3.

Primary note proof

Study-note section DSA29

Worst probes are floor($\log_2 n$)+1 for n>0: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list

• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Question sources

MOE Computer Science Exit Exam · Question 2-4

National Computer Science Exit Exam · Question 62

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 108

Question 120 | SM0016

What is the tightest listed worst-case time bound for ordinary quicksort when each partition may choose an extreme pivot?

A. O(n)

B. O(n^2)

C. O(2^n)

D. O(n log n)

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: B. O(n^2)

An extreme pivot leaves one empty part and one part with only a single element removed. Repeating that pattern makes partition work proportional to n+(n-1)+... rather than distributing it across balanced levels. The sum is quadratic. Balanced partitions explain the n log n alternative, but they are not guaranteed in this worst case. Exponential time is a looser upper bound, not the tightest listed one.

Why each choice is right or wrong

A - Incorrect: Linear time does not cover repeated partition scans.

B - Correct: Repeated extreme pivots produce quadratic total work.

C - Incorrect: Exponential is a much looser upper bound.

D - Incorrect: n log n requires suitably balanced or expected partitions.

Primary note proof

Study-note section DSA36

Choose a pivot, partition, then recursively sort the pieces. Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work. Expected n log n needs an input/random-pivot model. Random and median-of-three choices have different tradeoffs; neither is universally best.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Question sources

MOE Computer Science Exit Exam · Question 4-4

National Computer Science Exit Exam · Question 74

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 109

Question 121 | SM0043

Under universal hashing, one random family member is chosen for a table independently of a fixed key set. Which claims are false? Interpret C as an unconditional superiority claim. Select all that apply.

A. In universal hashing a hash function is selected randomly from a set of hash functions for each key

B. In universal hashing a hash function is selected from a set of hash functions independent of keys

C. A single fixed hash function is always better than universal hashing at reducing collisions.

D. Universal hashing controls collision probability without guaranteeing zero collisions.

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A, C: A. In universal hashing a hash function is selected randomly from a set of hash functions for each key; C. A single fixed hash function is always better than universal hashing at reducing collisions.

Universal hashing chooses one random function for the table and then uses it consistently, rather than selecting an unrelated function afresh for each key. Its purpose is to control collision probability for a fixed key set. It does not establish that a single fixed function is always better for avoiding collisions. Thus A and C are false, while independent table-level choice and probability control are the valid principles.

Why each choice is right or wrong

A - Correct: The chosen function is reused for the table rather than chosen afresh per key.

B - Incorrect: Independence from the fixed key set is the stated selection principle.

C - Correct: Unconditional fixed-function superiority does not follow and contradicts the randomized collision-control purpose.

D - Incorrect: Universal choice controls collision probability without promising no collisions.

Primary note proof

Study-note section DSA25

> Determinism means repeated equal keys get the same address for the chosen function. Uniformity means expected keys spread evenly. Universal hashing chooses one function randomly from a suitable family for the table, independently of a fixed key set; it does not choose a fresh unrelated function for each lookup. Random choice controls collision probability, not a promise of zero collisions.

Question sources

MOE Computer Science Exit Exam · Question 13-1

National Computer Science Exit Exam · Question 25

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 124

Question 122 | SM0050

In a tree data structure, if a node has no parent node, then the node is _____.

A. Parent node

B. Root node

C. External node

D. Internal node

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: Root node

The module defines root as a node without a parent, whereas leaves have no child and internal nodes have children. Evidence: assigned PDF pages 9.

Why each choice is right or wrong

A - Incorrect: Parent describes a relation to a child, not the absence of a parent. See assigned PDF 9.

B - Correct: Root is explicitly the parentless node. See assigned PDF 9.

C - Incorrect: External nodes are defined by having no child. See assigned PDF 9.

D - Incorrect: Internal nodes have at least one child, which does not imply parentlessness. See assigned PDF 9.

Primary note proof

Study-note section DSA18

> A rooted tree has one parentless root and one parent per other node. It is connected and acyclic, has n-1 edges, and one path between any pair. Depth counts root-to-node edges; height is maximum depth. A single-node tree therefore has height zero. Leaves have no children.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 3-13

> A tree is a set of nodes and edges that connect pairs of nodes that connect pairs of nodes. It is an abstract model of a hierarchical structure. Rooted tree has the following structure:
> • One node distinguished as root.
> • Every node C except the root is connected from exactly other node P. P is C's parent, and C is one of C's children.
> • There is a unique path from the root to the each node.
> • The number of edges in a path is the length of the path.
> Tree Terminologies:
> • Root: a node without a parent.
> • Internal node: a node with at least one child.
> • External (leaf) node: a node without a child.

Question sources

MOE Computer Science Exit Exam · Question 15-1

National Computer Science Exit Exam · Question 7

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 123

Question 123 | SM0053

Linked list data structure is _____ type of data structure.

A. Non-linear data structure

B. Fixed size data structure

C. Static data structure

D. Dynamic data structure

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: Dynamic data structure

Linked nodes are created using new and the module labels linked-list memory dynamic. The size is not fixed by a static array declaration. Evidence: assigned PDF pages 5, 6.

Why each choice is right or wrong

A - Incorrect: The supplied single linked list forms a linear chain. See assigned PDF 5, 6.

B - Incorrect: New nodes can be added at runtime, so size is not intrinsically fixed. See assigned PDF 5, 6.

C - Incorrect: Static describes the source array implementation. See assigned PDF 5, 6.

D - Correct: Dynamic allocation is the linked representation given. See assigned PDF 5, 6.

Primary note proof

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 5 | lines 3-8

> While adding a node to a list, the chain needs to be rearranged properly so that the each nodes are connected to each other. It is possible to add a node anywhere in the chain (at the start, middle or end). Let’s consider adding a node at the end of the list:
> Step 1: First we need to reserve a space for the new node using the keyword new.
> Node *temp = new *Node; // Reserving a space for temporary pointer which points to new node;

Data Structure and Algorithms module | PDF page 6 | lines 43-45

II. Linked list (dynamic memory)
The linked list implementation of stack is adding items at the end of the chain (Push item) and deleting the end node of the list (Pop item).

Question sources

MOE Computer Science Exit Exam · Question 15-4

National Computer Science Exit Exam · Question 95

National Computer Science Exit Exam

MOE Computer Science Exit Exam

MOE Computer Science Exit Exam · Question 27-4

National Computer Science Exit Exam · Question 99

Computer Science Exit Exam practice collection · Question 128

Question 124 | SM0064

Assume there is a line of patients waiting for service from a clinic. Seriously sick patients should get service first. Which data structure is the most appropriate for this real-world problem?

A. Tree

B. Queue

C. Stack

D. Priority Queue

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: Priority Queue

Give each patient a key representing medical urgency and serve by that priority. This adapts the module associated-key priority queue to the stated urgency rule. Evidence: assigned PDF pages 8, 7.

Why each choice is right or wrong

A - Incorrect: A tree alone does not define urgency-based service. See assigned PDF 8, 7.

B - Incorrect: A plain FIFO queue serves arrival order rather than illness severity. See assigned PDF 8, 7.

C - Incorrect: A stack gives recent arrivals precedence, regardless of severity. See assigned PDF 8, 7.

D - Correct: Associated priority keys represent how urgently each patient needs service. See assigned PDF 8, 7.

Primary note proof

Study-note section DSA17

A priority queue returns the smallest or largest priority key under its convention. Urgent patients may precede earlier arrivals. Insertions and extractions can interleave. Binary heaps support constant peek and logarithmic insert/extract.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 40-45

Priority Queue:- is a queue where each data has an associated key provided at insertion time.
Application of Queue
Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
• Print server- maintains a queue of print jobs
• Disk Driver- maintains a queue of disk input/output requests
• Task scheduler in multiprocessing system- maintains priority queues of processes

Data Structure and Algorithms module | PDF page 7 | lines 10-16

“A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

MOE Computer Science Exit Exam · Question 18-3

National Computer Science Exit Exam · Question 34

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 125

Question 125 | SM0076

For an input graph, Depth First Traversal (DFT) algorithm uses _____ data structure to traverse.

A. Priority queue

B. Tree

C. Queue

D. Stack

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: Stack

DSA42 says DFS recursively explores an unvisited neighbor to completion before returning. The recursive call stack stores return points; an explicit stack can store pending work instead. Both implement latest-pending-work-first order. A queue would change that discipline, while a priority queue or tree alone does not encode the required depth-first frontier behavior.

Why each choice is right or wrong

A - Incorrect: Priority service follows priority keys and does not guarantee finishing the latest active branch first.

B - Incorrect: A tree may represent traversal results, but it does not identify the standard DFS pending-work discipline.

C - Incorrect: FIFO queue order produces breadth-first pending work rather than the required latest-pending-work behavior.

D - Correct: A stack, explicit or supplied by recursion, preserves return points and latest-pending-work-first order.

Primary note proof

Study-note section DSA42

> Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours. An explicit stack can store pending work instead. Both implement latest-pending-work first; a queue would change that discipline. Marking prevents infinite repetition around graph cycles.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

> • Reversing Data and Reverse a List
> • Convert Decimal to Binary
> • Parsing - It is a logic that breaks into independent pieces for further processing
> • Backtracking
> Examples:
> 1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
> 2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.

Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

MOE Computer Science Exit Exam · Question 22-1

National Computer Science Exit Exam · Question 27

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 126

Question 126 | SM0082

Which statement is true about the pop() operation of Stack data structures?

A. pop() returns the end/top element but it doesn't delete that element

B. pop() removes the end/top element from the stack

C. pop() inserts a new element at the middle of stack

D. pop() inserts a new element at the beginning/bottom of stack

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: pop() removes the end/top element from the stack

The pop pseudocode reads the current top and decrements top, so the element is logically removed. Merely inspecting without deletion is peek. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Incorrect: This describes peek, not pop. See assigned PDF 6, 6.

B - Correct: Pop removes the current top by decreasing top. See assigned PDF 6, 6.

C - Incorrect: Pop does not insert, and stack insertion is only at top. See assigned PDF 6, 6.

D - Incorrect: Pop is removal, not bottom insertion. See assigned PDF 6, 6.

Primary note proof

Study-note section DSA11

Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

"A stack is an ordered list in which all insertions and deletions are made at one end, called the top". Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 6 | lines 24-42

I. Array implementation (static memory).
Push(n): Adding item at top
top is the current top of stack and n is its
maximum size
Begin
if (top == n) then stack is full;
else
top := top+1;
stack[top] := item;
end:
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;

Question sources

MOE Computer Science Exit Exam · Question 23-3

National Computer Science Exit Exam · Question 8

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 127

Question 127 | SM0210

A singly linked list of n nodes maintains only its head; there is no tail pointer or index. Appending traverses to the last node before linking a new node. Which listed bounds hold for its worst-case time? Select all that apply.

A. O(n)

B. Theta(1)

C. Theta(n)

D. O(1)

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A, C: A. O(n); C. Theta(n)

The append operation begins at the head and must follow the chain to locate its final node. That requires linear traversal, followed by a constant amount of link work. Its tight worst-case bound is therefore Theta(n), which also implies the upper bound O(n). Neither constant-time choice holds as the list grows. Maintaining a separate tail would change the operation cost, but that representation is explicitly excluded.

Why each choice is right or wrong

A - Correct: O(n) is a valid upper bound on the traversal.

B - Incorrect: The growing traversal prevents a constant tight bound.

C - Correct: Theta(n) is the tight traversal bound.

D - Incorrect: O(1) does not bound arbitrary head-only append traversal.

Primary note proof

Study-note section DSA09

> Prepend with new.next=head; head=new. To insert after a known predecessor p, set new.next=p.next before p.next=new. Appending with only head requires a tail traversal; maintaining tail avoids it. Delete the head by saving old=head, advancing head=head.next and releasing old in C++.

Study-note section DSA06

> For eventually nonnegative functions, f(n)=O(g(n)) means f(n)≤c g(n) for some positive c beyond a threshold k. Omega reverses the inequality; Theta requires both. Little-o means f/g tends to zero; little-omega means it grows without bound. These relations can describe best-case, average-case, worst-case or space functions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 5 | lines 13-24

Step 3: Then add the node at the end of the list. For this we need to check whether the list is empty or not. If list is empty the new node will be the first (Start) node, otherwise navigate forward starting form the first node till last node found and add it next to this node which make the new node the last node.
if (start == NULL) // If list is empty, then new node become start node
start = temp;
else {
temp2 = start; // list not empty! Thus temp2 starts from first node
while (temp2->next!= NULL) { // till the current last node found navigates forward
temp2 = temp2->next; // Move to next link in chain
}
temp2->next = temp; // new node (temp) appended at the end of the list

Data Structure and Algorithms module | PDF page 3 | lines 2-9

I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:
II. Big-Omega Notation (Ω): provides an asymptotic lower bound; f(n)= Ω(g (n)) means that the growth rate of f(n) is greater than or equal to g(n).
III. Theta Notation (Θ): If f(n)= Θ (g(n)), then g(n) is an asymptotically tight bound for f(n) i.e f(n) and g(n) have the same rate of growth.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 3-3

Computer Science Exit Exam practice collection · Question 138

Question 128 | SM0213

Which ordinary search method requires its input key sequence to be sorted before using comparisons to discard the half that cannot contain a target?

A. Hash search

B. Depth-first search

C. Binary search

D. Linear search

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: C. Binary search

Binary search compares a target with the midpoint of an ordered interval, then safely discards one half. That elimination depends on sorted order. Linear search can scan unsorted values, hashing selects addresses from keys rather than sorted positions, and depth-first search follows graph adjacency. The revised alternatives therefore have clear operational meanings, and Binary search uniquely matches the stated ordered-half-elimination requirement.

Why each choice is right or wrong

A - Incorrect: Hash lookup uses computed addresses rather than a sorted input interval.

B - Incorrect: Depth-first search follows adjacency, not ordered-array halves.

C - Correct: Binary search needs ordering to safely discard a comparison-selected half.

D - Incorrect: Linear scanning permits unsorted input.

Primary note proof

Study-note section DSA29

> On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1.

Study-note section DSA28

> Unsorted data is allowed; sorted data is allowed too.

Study-note section DSA25

> A hash table uses h(key) in 0..m-1 and stores only actual entries plus bucket capacity.

Study-note section DSA42

> Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Computer Science Exit Exam · Sene 2016 · Question 4-3

Computer Science Exit Exam practice collection · Question 139

Question 129 | SM0229

Given a binary search tree, which traversal type would print the values in the nodes in sorted order?

A. Pre-order

B. Pre-order and Post-order

C. In-order

D. Post-order

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: In-order

All left-subtree keys precede the root, and all right-subtree keys follow it. Recursively visiting left, parent, right therefore emits increasing keys. Evidence: assigned PDF pages 9.

Why each choice is right or wrong

A - Incorrect: Preorder prints root before smaller left keys. See assigned PDF 9, 9.

B - Incorrect: Including preorder cannot guarantee increasing order. See assigned PDF 9, 9.

C - Correct: Inorder matches left keys < root < right keys recursively. See assigned PDF 9, 9.

D - Incorrect: Postorder prints the root after larger right keys. See assigned PDF 9, 9.

Primary note proof

Study-note section DSA19

With distinct keys, every key in a node's left subtree is smaller and every key in its right subtree is larger. Immediate-child comparisons alone are insufficient: root 10, left child 5, and 5's right child 12 violate the root's restriction. Duplicates require a separate count or placement policy.

Study-note section DSA20

Preorder visits node,left,right; inorder left,node,right; postorder left,right,node. Apply recursively; null children do nothing. Inorder returns sorted keys only for a BST. Preorder supports copying, postorder supports deleting children before parents, and level order uses a FIFO queue.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Data Structure and Algorithms module | PDF page 9 | lines 40-44

III. Tree Traversing:
Binary search tree can be traversed in three ways.
a. Pre order traversal - traversing binary tree in the order of parent, left and right.
b. Inorder traversal - traversing binary tree in the order of left, parent and right.
c. Postorder traversal - traversing binary tree in the order of left, right and parent.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 9-4

Computer Science Exit Exam practice collection · Question 137

Question 130 | SM0240

If a sequence of operations: push(1), push(2), pop, push(1), push(2), pop, pop, pop, push(2), pop are performed on a stack, the sequence of popped values is:

A. 2,1,2,2,1

B. 2,2,1,1,2

C. 2,1,2,2,2

D. 2,2,1,2,2

Hint

Trace the initial state and update only the fields affected by each permitted operation.

Answer B: 2,2,1,1,2

Trace bottom-to-top: [1], [1,2], pop 2; [1,1], [1,1,2], pop 2; pop 1; pop 1; [2], pop 2. Popped sequence is 2,2,1,1,2. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Incorrect: The second pop is 2, not 1. See assigned PDF 6.

B - Correct: Exactly matches all five top removals. See assigned PDF 6.

C - Incorrect: The second pop is again 2, not 1. See assigned PDF 6.

D - Incorrect: The fourth pop removes the original bottom 1, not 2. See assigned PDF 6.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA12

> Write the stack bottom-to-top after every instruction and record popped values separately. Push 1, push 2, pop emits 2 and leaves [1]. Push 1, push 2, pop, pop, pop then emits 2,1,1; push 2, pop emits 2. The complete output is 2,2,1,1,2.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science Exit Exam · Sene 2016 · Question 13-2

Computer Science Exit Exam practice collection · Question 140

Question 131 | SM0279

Which of the following is FALSE in Data Structures and Algorithms?

A. A tree contains a cycle

B. Tree is a non-linear data structure

C. A tree is a connected graph

D. A tree with n nodes contains (n-1) edges

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A: A tree contains a cycle

Unique root paths and one parent per nonroot node exclude cycles. The n-1 nonroot nodes each contribute one parent edge, making the rooted tree connected with n-1 edges. Evidence: assigned PDF pages 9, 2.

Why each choice is right or wrong

A - Correct: A cycle contradicts the source unique-path parent hierarchy. See assigned PDF 9, 2.

B - Incorrect: Trees are explicitly classified as nonlinear. See assigned PDF 9, 2.

C - Incorrect: Every node is reachable from the root along a unique path, so the tree is connected. See assigned PDF 9, 2.

D - Incorrect: One parent edge for each nonroot node yields n-1 edges. See assigned PDF 9, 2.

Primary note proof

Study-note section DSA18

> A rooted tree has one parentless root and one parent per other node. It is connected and acyclic, has n-1 edges, and one path between any pair. Depth counts root-to-node edges; height is maximum depth. A single-node tree therefore has height zero. Leaves have no children.

Study-note section DSA01

> An abstract data type (ADT) defines values and operations; representation supplies implementation. Arrays or linked nodes can implement the same LIFO stack. Abstraction keeps relevant details: patient urgency and arrival order matter for service; hair colour does not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 3-13

> A tree is a set of nodes and edges that connect pairs of nodes that connect pairs of nodes. It is an abstract model of a hierarchical structure. Rooted tree has the following structure:
> • One node distinguished as root.
> • Every node C except the root is connected from exactly other node P. P is C's parent, and C is one of C's children.
> • There is a unique path from the root to the each node.
> • The number of edges in a path is the length of the path.
> Tree Terminologies:
> • Root: a node without a parent.
> • Internal node: a node with at least one child.
> • External (leaf) node: a node without a child.

Data Structure and Algorithms module | PDF page 2 | lines 15-18

a. Linear data structure: data items stores in sequence
Example: Array, stack and Queue,
b. Non-Linear data structures : in which there is order of data items
Example: Trees and Graphs

Question sources

Computer Science Exit Exam · Sene 2016 · Question 26-1

Computer Science Exit Exam practice collection · Question 136

Question 132 | SM0282

A fixed-capacity circular array queue uses front and size indices, modulo slot reuse, and no shifting or resizing. Under unit-cost index arithmetic, what is the tightest listed worst-case pair for enqueue and dequeue, including full/empty checks?

A. O(n), O(1)

B. O(1), O(n)

C. O(1), O(1)

D. O(n), O(n)

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: C. O(1), O(1)

Enqueue computes a slot from front and size, writes one item and updates the count. Dequeue reads the front slot, advances front modulo capacity and decreases the count. Full or empty checks also inspect only constant-size state. No operation shifts or resizes the array, so each uses constant work in the stated model. The linear alternatives are looser bounds rather than the requested tight pair.

Why each choice is right or wrong

A - Incorrect: Linear enqueue is looser than the constant cost of this implementation.

B - Incorrect: Linear dequeue is looser than its constant index update cost.

C - Correct: Both endpoint operations use constant work.

D - Incorrect: Both linear bounds are looser than the requested tight pair.

Primary note proof

Study-note section DSA15

A simple noncircular array queue may report overflow when rear reaches the final index despite released slots at the front. A circular queue reuses slots with modulo arithmetic. Use a count convention: front points to the next removal, size counts items, and insertion slot is (front+size)%C. Empty means size==0; full means size==C.

Study-note section DSA15

Dequeue reads A[front], sets front=(front+1)%C and decreases size. An alternative reserves one unused slot and uses next(rear)==front for full; its empty test and capacity differ. State the convention before tracing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 18-23

I. Simple Array implementation (static memory).
A simple array structure can be used to implement enqueue & dequeue operations. Thus item will
be added at the end of the array whereas dequeue operation removes items from the front. In simple
array implementation, to add item it checks whether there is room at the rear or not. Therefore, even though there are free released spaces at the front ot the array the queue implementation might
generate overflow message. This is the key drawback to use simple array to implement Queue.

Data Structure and Algorithms module | PDF page 7 | lines 24-26

II. Circular array implementation of Queue Operations
To overcome the above drawback of simple array implementation a circular array can be used, in
which freed spaces are re-used to store data.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 27-1

Computer Science Exit Exam practice collection · Question 141

Question 133 | SS115

Depending on the organization of the elements which of the following approach Is a non-linear data structure?

A. Queues

B. Linked-list

C. Tree

D. Stack

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: Tree

The module classifies trees as nonlinear, while queues and stacks are linear sequences and a linked list is a next-pointer chain. Evidence: assigned PDF pages 2, 4, 6.

Why each choice is right or wrong

A - Incorrect: Queue is an explicit linear example. See assigned PDF 2, 4, 6.

B - Incorrect: A linked list is a chain with one next node, not branching hierarchy. See assigned PDF 2, 4, 6.

C - Correct: Tree is an explicit nonlinear example. See assigned PDF 2, 4, 6.

D - Incorrect: Stack is an explicit linear example. See assigned PDF 2, 4, 6.

Primary note proof

Study-note section DSA01

> An abstract data type (ADT) defines values and operations; representation supplies implementation. Arrays or linked nodes can implement the same LIFO stack. Abstraction keeps relevant details: patient urgency and arrival order matter for service; hair colour does not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 15-18

> a. Linear data structure: data items stores in sequence
> Example: Array, stack and Queue,
> b. Non-Linear data structures : in which there is order of data items
> Example: Trees and Graphs

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
> to the chain.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some

useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science model exam collection · Question 22

Question 134 | SS149

For this question, an indirect queue application means using FIFO as auxiliary storage inside a graph traversal algorithm, rather than directly modeling a waiting line or scheduling service. Which listed description fits?

A. Operating systems schedule jobs (with equal priority) in the order of arrival (e.g., a print queue).

B. Simulation of real-world queues such as lines at a ticket counter or any other first-come first-served scenario requires a queue.

C. Auxiliary data structure for algorithms

D. Multiprogramming

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: C. Auxiliary data structure for algorithms

The supplied convention distinguishes an algorithm's internal work storage from a real-world waiting-line model. BFS uses a queue to hold its discovered but not yet explored vertices, so auxiliary algorithm storage is the matching category. Equal-priority job scheduling and ticket-line simulation directly model service waiting. Multiprogramming is a broader operating-system setting and does not specifically name the internal graph-traversal role requested here.

Why each choice is right or wrong

A - Incorrect: Arrival-order print scheduling directly models a service queue.

B - Incorrect: Ticket-line simulation directly models real-world waiting.

C - Correct: An algorithmic frontier queue is auxiliary storage inside graph traversal.

D - Incorrect: Multiprogramming is a broad execution setting, not the specified internal traversal role.

Primary note proof

Study-note section DSA41

Mark the source when enqueuing it. Repeatedly dequeue a vertex, examine its neighbours, and enqueue each unvisited neighbour while marking it. Store predecessor[v] when discovering v if a path must be reconstructed. FIFO ensures vertices at distance d are explored before later distance layers.

Study-note section DSA14

FIFO suits equal-priority printer jobs, shared-resource waiting, asynchronous buffers and level-order exploration. A play sequence can also use a queue. Distinguish airline seat records from waiting requests; an application name alone does not force one representation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 41-48

Application of Queue
Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
• Print server- maintains a queue of print jobs
• Disk Driver- maintains a queue of disk input/output requests
• Task scheduler in multiprocessing system- maintains priority queues of processes
• Telephone calls in a busy environment -maintains a queue of telephone calls
• Simulation of waiting line- models real life queues (e.g. supermarkets checkout, customer service line (Bank, custom, airport etc))

Question sources

Computer Science model exam collection · Question 56

Question 135 | SS158

A queue in which the item most recently added is always the first one out refers to:

A. FIFO queue

B. Real time queue

C. Priority queue

D. LIFO queue

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: LIFO queue

Most recently added first removed is LIFO by definition. The name LIFO queue here describes stack discipline rather than the ordinary FIFO queue. Evidence: assigned PDF pages 6, 7, 8.

Why each choice is right or wrong

A - Incorrect: FIFO removes the earliest insertion. See assigned PDF 6, 7, 8.

B - Incorrect: Real time alone does not specify last-in removal. See assigned PDF 6, 7, 8.

C - Incorrect: Priority depends on keys, not necessarily insertion recency. See assigned PDF 6, 7, 8.

D - Correct: LIFO exactly names last-in-first-out service. See assigned PDF 6, 7, 8.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 8 | lines 40-45

> Priority Queue:- is a queue where each data has an associated key provided at insertion time.
> Application of Queue
> Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
> • Print server- maintains a queue of print jobs
> • Disk Driver- maintains a queue of disk input/output requests
> • Task scheduler in multiprocessing system- maintains priority queues of processes

Question sources

Computer Science model exam collection · Question 65

Question 136 | AM0394

Which data structure follows the LIFO (Last-In-First-Out) principle?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A: Stack

The stack top is both insertion and removal end, so the most recent unremoved insertion leaves first. Evidence: assigned PDF pages 6, 7, 4, 9.

Why each choice is right or wrong

A - Correct: Stack is explicitly LIFO. See assigned PDF 6, 7, 4, 9.

B - Incorrect: Queue is FIFO. See assigned PDF 6, 7, 4, 9.

C - Incorrect: A linked list is a representation with no mandatory service discipline. See assigned PDF 6, 7, 4, 9.

D - Incorrect: A binary tree defines children, not insertion/removal chronology. See assigned PDF 6, 7, 4, 9.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 7 | lines 10-16

“A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Question sources

Computer Science MCQ practice collection · Question 394

Computer Science MCQ practice collection · Question 414

Question 137 | AM0395

Which data structure follows the FIFO (First-In-First-Out) principle?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: Queue

FIFO is explicitly the queue policy: enqueue at rear and dequeue at front. Evidence: assigned PDF pages 7, 6.

Why each choice is right or wrong

A - Incorrect: Stack is LIFO. See assigned PDF 7, 6.

B - Correct: Queue preserves insertion order for removal. See assigned PDF 7, 6.

C - Incorrect: A linked list can implement a queue but does not inherently impose FIFO. See assigned PDF 7, 6.

D - Incorrect: Binary-tree structure does not impose FIFO. See assigned PDF 7, 6.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science MCQ practice collection · Question 395

Computer Science MCQ practice collection · Question 415

Question 138 | AM0396

Which listed representation permits insertion and removal at either endpoint in constant time? Use ordinary stack and FIFO queue interfaces, and interpret C as a doubly linked list maintaining both head and tail.

A. Stack

B. Queue

C. Doubly linked list with head and tail

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: C. Doubly linked list with head and tail

A doubly linked list stores predecessor and successor links, while head and tail provide direct access to both endpoints. Inserting or removing there updates a fixed number of links, including the empty and singleton cases. Ordinary stack and FIFO queue interfaces expose fewer endpoint operations, and a binary tree has no corresponding two-end sequence guarantee. The explicit linked-list variant is essential because singly linked tail deletion needs a predecessor traversal.

Why each choice is right or wrong

A - Incorrect: A stack exposes insertion/removal at one top only.

B - Incorrect: An ordinary FIFO queue does not permit all four endpoint operations.

C - Correct: A doubly linked list with both endpoint pointers supports constant-time operations at each end.

D - Incorrect: A binary tree has no two-end sequence-operation guarantee.

Primary note proof

Study-note section DSA10

> Doubly linked nodes add prev; head and tail enable constant-time operations at both ends. Singly linked tail deletion still needs its predecessor. A circular tail links back to head; stop traversal on returning to the start, handling empty separately.

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
> to the chain.

Data Structure and Algorithms module | PDF page 8 | lines 38-40

> Deque (pronounced as Deck): is a Double Ended Queue in which insertion and deletion can occur at either end.
> Priority Queue:- is a queue where each data has an associated key provided at insertion time.

Question sources

Computer Science MCQ practice collection · Question 396

Question 139 | AM0397

Which data structure represents a hierarchical structure with nodes and edges?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer D: Binary tree

A tree is expressly an abstract model of a hierarchical structure, and a binary tree is its at-most-two-children specialization. Evidence: assigned PDF pages 9.

Why each choice is right or wrong

A - Incorrect: Stack is a one-ended linear list. See assigned PDF 9, 9.

B - Incorrect: Queue is a FIFO linear list. See assigned PDF 9, 9.

C - Incorrect: A singly linked list is a chain rather than a branching hierarchy. See assigned PDF 9, 9.

D - Correct: Binary tree matches the hierarchy with nodes and edges. See assigned PDF 9, 9.

Primary note proof

Study-note section DSA18

> A rooted tree has one parentless root and one parent per other node. It is connected and acyclic, has n-1 edges, and one path between any pair. Depth counts root-to-node edges; height is maximum depth. A single-node tree therefore has height zero. Leaves have no children.

Study-note section DSA01

> Trees model hierarchies; graphs model general connections.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 3-13

> A tree is a set of nodes and edges that connect pairs of nodes that connect pairs of nodes. It is an abstract model of a hierarchical structure. Rooted tree has the following structure:
> • One node distinguished as root.
> • Every node C except the root is connected from exactly other node P. P is C's parent, and C is one of C's children.
> • There is a unique path from the root to the each node.
> • The number of edges in a path is the length of the path.
> Tree Terminologies:

• Root: a node without a parent.
• Internal node: a node with at least one child.
• External (leaf) node: a node without a child.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Question sources

Computer Science MCQ practice collection · Question 397

Question 140 | AM0398

Which data structure is best suited for implementing a backtracking algorithm?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A: Stack

Backtracking is explicitly listed as a stack application: the latest pending decision can be undone first using pop. Evidence: assigned PDF pages 7, 6.

Why each choice is right or wrong

A - Correct: LIFO naturally returns to the most recent unfinished decision. See assigned PDF 7, 6.

B - Incorrect: FIFO would return to oldest pending work. See assigned PDF 7, 6.

C - Incorrect: A generic linked list does not prescribe backtracking order. See assigned PDF 7, 6.

D - Incorrect: A binary tree can represent choices but does not itself maintain a LIFO return path. See assigned PDF 7, 6.

Primary note proof

Study-note section DSA12

Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

• Reversing Data and Reverse a List
• Convert Decimal to Binary
• Parsing - It is a logic that breaks into independent pieces for further processing
• Backtracking
Examples:
1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Data Structure and Algorithms module | PDF page 6 | lines 15-22

“A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science MCQ practice collection · Question 398

Question 141 | AM0399

A dispatcher keeps pending jobs in a sequence. Urgent jobs are added at the front, ordinary jobs at the back; serving removes the front job, and cancellation removes the last job. Which listed representation supports every operation in worst-case constant time as the sequence grows? Use ordinary stack and FIFO queue interfaces, with no additional indexes or auxiliary structures.

A. Stack

B. Queue

C. Doubly linked list with head and tail

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: C. Doubly linked list with head and tail

Urgent arrival inserts at the head, ordinary arrival appends at the tail, service removes the head, and cancellation removes the tail. A doubly linked list with both endpoint pointers updates a fixed number of links for each action; its prev link identifies the predecessor needed for tail removal. A stack exposes only one top, a FIFO queue lacks last-job cancellation and front insertion, and a generic binary tree provides no such endpoint-cost guarantee.

Why each choice is right or wrong

A - Incorrect: One stack top cannot directly supply both service-at-front and cancellation-at-back operations.

B - Incorrect: An ordinary FIFO queue permits rear insertion and front removal, but not urgent front insertion or last-job removal.

C - Correct: Head and tail permit direct endpoint access, and prev links make last-job removal constant time.

D - Incorrect: A generic binary tree has no guarantee for all four ordered-sequence endpoint updates.

Primary note proof

Study-note section DSA10

> Doubly linked nodes add prev; head and tail enable constant-time operations at both ends. Singly linked tail deletion still needs its predecessor. A circular tail links back to head; stop traversal on returning to the start, handling empty separately.

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node

added
to the chain.

Data Structure and Algorithms module | PDF page 5 | lines 13-24

Step 3: Then add the node at the end of the list. For this we need to check whether the list is empty or not. If list is empty the new node will be the first (Start) node, otherwise navigate forward starting form the first node till last node found and add it next to this node which make the new node the last node.
if (start == NULL) // If list is empty, then new node become start node
start = temp;
else {
temp2 = start; // list not empty! Thus temp2 starts from first node
while (temp2->next!= NULL) { // till the current last node found navigates forward
temp2 = temp2->next; // Move to next link in chain
}
temp2->next = temp; // new node (temp) appended at the end of the list

Data Structure and Algorithms module | PDF page 6 | lines 4-13

2. Make temp1 point to whatever the start pointer is pointing to.
3. If the next pointer of what temp1 indicates is NULL, then we've found the last node of the list, so
jump to step 7.
4. Make another pointer, temp2, point to the current node in the list.
5. Make temp1 point to the next item in the list.
6. Go to step 3.
7. If you get this far, then the temporary pointer, temp1, should point to the last item in the list and
the other temporary pointer, temp2, should point to the last-but-one item.
8. Delete the node pointed to by temp1.
9. Mark the next pointer of the node pointed to by temp2 as NULL - it is the new last node.

Data Structure and Algorithms module | PDF page 8 | lines 38-40

Deque (pronounced as Deck): is a Double Ended Queue in which insertion and deletion can occur at either end.
Priority Queue:- is a queue where each data has an associated key provided at insertion time.

Question sources

Computer Science MCQ practice collection · Question 399

Question 142 | AM0400

Which listed structures admit both array-based and linked-node representations? Links may be array indices or pointers, and binary-tree representations include the complete binary-tree case. Select all that apply.

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A, B, C, D: A. Stack; B. Queue; C. Linked list; D. Binary tree

Stacks and queues have both array and linked-node implementations. A linked list can store its next links as indices into an array or as pointers between allocated nodes. A binary tree can use linked child references, while the complete binary-tree case has the indexed heap representation. The question concerns possible representations, so all four qualify without requiring one universal layout for every instance.

Why each choice is right or wrong

A - Correct: A stack can use an array top or a linked head.

B - Correct: A queue can use circular slots or linked endpoint nodes.

C - Correct: List links may be indices or node references.

D - Correct: Binary trees have linked-node forms and indexed complete-tree forms.

Primary note proof

Study-note section DSA01

> Arrays or linked nodes can implement the same LIFO stack.

Study-note section DSA14

> A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head.

Study-note section DSA15

> A circular queue reuses slots with modulo arithmetic.

Study-note section DSA08

> An array-linked list still follows links unless separately position-indexed.

Study-note section DSA23

> Zero-based array representation: root index 0, left=2i+1, right=2i+2, parent=floor((i-1)/2) for i>0.

Study-note section DSA-S03

A linked binary-tree node stores a payload and two child pointers, one for the left child and one for the right child.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 24-42

I. Array implementation (static memory).
Push(n): Adding item at top
top is the current top of stack and n is its
maximum size
Begin
if (top == n) then stack is full;
else
top := top+1;
stack[top] := item;
end:
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;

Data Structure and Algorithms module | PDF page 6 | lines 43-45

II. Linked list (dynamic memory)
The linked list implementation of stack is adding items at the end of the chain (Push item) and deleting the end node of the list (Pop item).

Data Structure and Algorithms module | PDF page 7 | lines 18-23

I. Simple Array implementation (static memory).
A simple array structure can be used to implement enqueue & dequeue operations. Thus item will
be added at the end of the array whereas dequeue operation removes items from the front. In simple
array implementation, to add item it checks whether there is room at the rear or not. Therefore, even though there are free released spaces at the front ot the array the queue implementation might
generate overflow message. This is the key drawback to use simple array to implement Queue.

Data Structure and Algorithms module | PDF page 8 | lines 35-37

III. Linked list implementation of enqueue and dequeue operations
Enqueue- is inserting a node at the end of a linked list
Dequeue- is deleting the first node in the list

Question sources

Computer Science MCQ practice collection · Question 400

Computer Science MCQ practice collection · Question 418

Question 143 | AM0401

A character reader supplies a word from left to right. After reading finishes, an output routine must remove one character at a time from a single auxiliary ADT to display the word right to left. It may use only the ADT's standard insertion/removal interface and must preserve the original input. Which ADT directly supplies the needed removal order?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A: A. Stack

Insert the characters in the order they are read. The required output starts with the last character inserted, then continues toward the earliest one, so a stack's LIFO interface directly supplies the reversal. A queue would preserve the original order. A linked list could represent a stack, but its storage links alone do not prescribe LIFO; a generic tree likewise has no matching removal contract.

Why each choice is right or wrong

A - Correct: Repeated stack removals produce the characters in reverse insertion order.

B - Incorrect: FIFO removal would reproduce the original character order.

C - Incorrect: A list is a possible representation but does not itself prescribe the required removal policy.

D - Incorrect: A generic tree does not define the requested reverse-insertion removal contract.

Primary note proof

Study-note section DSA12

Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Study-note section DSA11

Stack policy is LIFO, equivalently FILO.

Study-note section DSA11

Linked stacks achieve constant time by inserting and deleting at the head; using a singly linked tail for pop would require a predecessor traversal.

Study-note section DSA14

An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served.

Study-note section DSA45

Specify whether a question asks which structure can implement something or which is best under stated costs. ADTs do not force one representation, and serial-access memory is not inherently LIFO.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

• Reversing Data and Reverse a List
• Convert Decimal to Binary
• Parsing - It is a logic that breaks into independent pieces for further processing
• Backtracking
Examples:
1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Data Structure and Algorithms module | PDF page 6 | lines 15-22

“A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science MCQ practice collection · Question 401

Computer Science MCQ practice collection · Question 424

Question 144 | AM0402

Which data structure is commonly used for implementing function call stacks in programming languages?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A: Stack

The source explicitly states that a memory stack is used in function calls including recursion, keeping the most recent active call on top. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Correct: Nested calls return in LIFO order through a stack. See assigned PDF 6, 6.

B - Incorrect: Queue FIFO order would not reflect nested returns. See assigned PDF 6, 6.

C - Incorrect: Linked nodes can store frames but the controlling ADT is still a stack. See assigned PDF 6, 6.

D - Incorrect: A binary tree does not provide the specified call-return discipline. See assigned PDF 6, 6.

Primary note proof

Study-note section DSA12

> Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 46-52

> Applications of Stack
> • It is very useful to evaluate arithmetic expressions. (Postfix Expressions)
> • Infix to Postfix Transformation
> • It is useful during the execution of recursive programs
> • A Stack is useful for designing the compiler in operating system to store local variables inside a function block.
> • A stack (memory stack) can be used in function calls including recursion.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

"A stack is an ordered list in which all insertions and deletions are made at one end, called the top". Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science MCQ practice collection · Question 402

Question 145 | AM0403

A message buffer accepts new items at its rear and delivers the oldest waiting item from its front. Which service ADT can maintain this contract using either a circular array or linked head-and-tail nodes? Choose the ADT rather than an underlying storage container.

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: B. Queue

The oldest waiting message must leave first, while new arrivals join the rear. This is the ordinary queue service contract. The note supplies both a circular-array representation that reuses slots and a linked representation that appends at the tail and removes at the head. A stack serves the newest item first; a generic linked list or tree does not independently specify this FIFO policy.

Why each choice is right or wrong

A - Incorrect: A stack's latest-first removal violates oldest-message-first delivery.

B - Correct: A queue has the stated rear-insertion/front-removal contract and both cited implementations.

C - Incorrect: Linked nodes can store the queue but do not by themselves define its service policy.

D - Incorrect: A binary tree does not inherently define this FIFO message-buffer interface.

Primary note proof

Study-note section DSA15

A simple noncircular array queue may report overflow when rear reaches the final index despite released slots at the front. A circular queue reuses slots with modulo arithmetic. Use a

count convention: front points to the next removal, size counts items, and insertion slot is (front+size)%C. Empty means size==0; full means size==C.

Study-note section DSA14

An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 24-26

II. Circular array implementation of Queue Operations
To overcome the above drawback of simple array implementation a circular array can be used, in
which freed spaces are re-used to store data.

Data Structure and Algorithms module | PDF page 8 | lines 35-37

III. Linked list implementation of enqueue and dequeue operations
Enqueue- is inserting a node at the end of a linked list
Dequeue- is deleting the first node in the list

Question sources

Computer Science MCQ practice collection · Question 403

Question 146 | AM0404

Which data structure has a First-In-First-Out (FIFO) property?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: Queue

First-in-first-out is the explicit queue property. Inserting at rear and removing at front preserves the oldest pending element. Evidence: assigned PDF pages 7.

Why each choice is right or wrong

A - Incorrect: Stack is last-in-first-out. See assigned PDF 7.

B - Correct: Queue has FIFO semantics. See assigned PDF 7.

C - Incorrect: A linked-list representation alone imposes no FIFO restriction. See assigned PDF 7.

D - Incorrect: A binary-tree hierarchy is not arrival-order access. See assigned PDF 7.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Computer Science MCQ practice collection · Question 404

Question 147 | AM0405

Which data structure is suitable for implementing a breadth-first search algorithm?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer B: Queue

DSA41 enqueues discovered neighbors and dequeues the next pending vertex. This FIFO discipline preserves distance-layer ordering. A linked list can store the queue but does not by itself state the policy. Stack service is latest-first, while a binary tree is not the standard frontier discipline. Queue is therefore the specific suitable ADT among the choices.

Why each choice is right or wrong

A - Incorrect: Stack order selects the latest pending vertex and does not preserve the BFS layer invariant.

B - Correct: Queue FIFO ordering processes earlier distance layers before later discoveries, as BFS requires.

C - Incorrect: A linked list may represent a queue, but linked-list links alone specify no BFS service order.

D - Incorrect: A binary tree does not supply the standard FIFO pending-vertex discipline used by breadth-first search.

Primary note proof

Study-note section DSA41

> Mark the source when enqueuing it. Repeatedly dequeue a vertex, examine its neighbours, and enqueue each unvisited neighbour while marking it. Store predecessor[v] when discovering v if a path must be reconstructed. FIFO ensures vertices at distance d are explored before later distance layers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.

Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Computer Science MCQ practice collection · Question 405

Computer Science MCQ practice collection · Question 426

Question 148 | AM0406

Which listed structures can create nodes using dynamic memory allocation in a linked-node implementation? Select all that apply.

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A, B, C, D: A. Stack; B. Queue; C. Linked list; D. Binary tree

Dynamic node allocation is a representation choice rather than a property exclusive to one ADT. A linked stack can allocate a node for each push, a linked queue for each enqueue, and a linked list for insertion. A pointer-based binary tree can allocate a leaf when attaching it. Array implementations also exist, but their existence does not prevent any of these structures from using dynamically allocated nodes.

Why each choice is right or wrong

A - Correct: A linked stack can allocate each new head node.

B - Correct: A linked queue can allocate each new rear node.

C - Correct: A linked list can grow by allocating nodes as needed.

D - Correct: A linked binary tree can allocate an inserted leaf.

Primary note proof

Study-note section DSA-S03

A linked binary-tree node stores a payload and two child pointers, one for the left child and one for the right child. A null child pointer means that child is absent; a root pointer gives access to the whole tree. In a dynamically allocated version, insertion can allocate a node and attach it through the appropriate child link.

Study-note section DSA08

Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Study-note section DSA11

Linked stacks achieve constant time by inserting and deleting at the head;

Study-note section DSA14

A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 5 | lines 3-8

While adding a node to a list, the chain needs to be rearranged properly so that the each nodes are connected to each other. It is possible to add a node anywhere in the chain (at the start, middle or end). Let's consider adding a node at the end of the list:
Step 1: First we need to reserve a space for the new node using the keyword new.
Node *temp = new *Node; // Reserving a space for temporary pointer which points to new node;

Data Structure and Algorithms module | PDF page 6 | lines 43-45

II. Linked list (dynamic memory)
The linked list implementation of stack is adding items at the end of the chain (Push item) and deleting the end node of the list (Pop item).

Data Structure and Algorithms module | PDF page 8 | lines 35-37

III. Linked list implementation of enqueue and dequeue operations
Enqueue- is inserting a node at the end of a linked list
Dequeue- is deleting the first node in the list

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Question sources

Computer Science MCQ practice collection · Question 406

Question 149 | AM0407

Which structures admit an array-based representation? Array slots may contain payloads and integer links; for binary trees include complete trees stored with child-index formulas. Select all that apply.

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer A, B, C, D: A. Stack; B. Queue; C. Linked list; D. Binary tree

An array stack uses a top index, while a circular queue uses front and size indices. A linked list can store its links as indices into an array rather than as pointers to separately allocated nodes. Complete binary trees also fit the indexed heap layout. Since the question asks whether representations exist and removes the subjective ease ranking, each of the four listed structures qualifies.

Why each choice is right or wrong

A - Correct: An array top index implements a stack.

B - Correct: Modulo-indexed slots implement a circular queue.

C - Correct: Array indices can encode list links.

D - Correct: Complete binary trees use the indexed heap layout.

Primary note proof

Study-note section DSA11

> With zero-based occupied slots, top=-1 is empty and top=C-1 is full.

Study-note section DSA15

> insertion slot is (front+size)%C.

Study-note section DSA08

> An array-linked list still follows links unless separately position-indexed.

Study-note section DSA23

> Zero-based array representation: root index 0, left=2i+1, right=2i+2, parent=floor((i-1)/2) for i>0. One-based representation uses children 2i,2i+1 and parent floor(i/2). A child exists only if its index is inside the current heap size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 24-42

I. Array implementation (static memory).
Push(n): Adding item at top
top is the current top of stack and n is its
maximum size
Begin
if (top == n) then stack is full;
else
top := top+1;
stack[top] := item;
end:
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;

Data Structure and Algorithms module | PDF page 7 | lines 18-23

I. Simple Array implementation (static memory).
A simple array structure can be used to implement enqueue & dequeue operations. Thus item will
be added at the end of the array whereas dequeue operation removes items from the front. In simple
array implementation, to add item it checks whether there is room at the rear or not. Therefore, even though there are free released spaces at the front ot the array the queue implementation might
generate overflow message. This is the key drawback to use simple array to implement Queue.

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.

Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Question sources

Computer Science MCQ practice collection · Question 407

Question 150 | AM0408

Which listed representation provides direct next and previous links for walking a linear sequence forward and backward without retracing from its head? Use ordinary stack/queue interfaces and a binary tree without parent or traversal-thread links.

A. Stack

B. Queue

C. Doubly linked list

D. Binary tree

Hint

Compare the required operations, access order and representation metadata with each option.

Answer C: C. Doubly linked list

A doubly linked list retains the forward next link and adds a prev link to each node. Following the corresponding link moves directly to a neighboring node in either direction. Ordinary stack and queue interfaces do not expose this bidirectional traversal contract. A child-linked binary tree without parent or traversal threads also does not provide the specified previous/next linear-sequence links. The concrete list variant makes the guarantee precise.

Why each choice is right or wrong

A - Incorrect: A stack interface exposes its top, not a bidirectional traversal.

B - Incorrect: A queue interface exposes FIFO endpoints, not neighboring prev/next traversal.

C - Correct: Doubly linked nodes provide next and prev links.

D - Incorrect: The specified child-only tree lacks the requested linear previous/next links.

Primary note proof

Study-note section DSA10

Doubly linked nodes add prev; head and tail enable constant-time operations at both ends. Singly linked tail deletion still needs its predecessor. A circular tail links back to head; stop traversal on returning to the start, handling empty separately.

Study-note section DSA08

A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Question sources

Computer Science MCQ practice collection · Question 408

Question 151 | AM0409

Which specified representations provide direct constant-time inspection of both first and last items in a nonempty sequence? A is a head-only linked stack, B a linked queue with front and rear pointers, C a linked list with head and tail, and D a child-linked binary tree with no cached endpoints. Select all that apply.

A. Head-only linked stack

B. Linked queue with front and rear

C. Linked list with head and tail

D. Child-linked binary tree without endpoint caches

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B, C: B. Linked queue with front and rear; C. Linked list with head and tail

Maintaining references to both endpoints makes their items directly available without traversal. The specified linked queue retains front and rear, and the specified list retains head and tail, so both provide constant-time inspection. A head-only linked stack must follow links to reach its oldest item, and a generic child-linked tree without endpoint caches supplies no direct first/last guarantee. The answer follows the representations, not the generic names alone.

Why each choice is right or wrong

A - Incorrect: A head-only stack lacks a direct reference to its oldest node.

B - Correct: Front and rear pointers directly expose both endpoint items.

C - Correct: Head and tail pointers directly expose both endpoint items.

D - Incorrect: The unaugmented tree has no cached traversal endpoints.

Primary note proof

Study-note section DSA14

> A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head.

Study-note section DSA10

> Doubly linked nodes add prev; head and tail enable constant-time operations at both ends.

Study-note section DSA11

> Linked stacks achieve constant time by inserting and deleting at the head;

Study-note section DSA-S03

> A linked binary-tree node stores a payload and two child pointers, one for the left child and one for the right child.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow

to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 5 | lines 13-24

Step 3: Then add the node at the end of the list. For this we need to check whether the list is empty or not. If list is empty the new node will be the first (Start) node, otherwise navigate forward starting form the first node till last node found and add it next to this node which make the new node the last node.

```
if (start == NULL) // If list is empty, then new node become start node
start = temp;
else {
temp2 = start; // list not empty! Thus temp2 starts from first node
while (temp2->next!= NULL) { // till the current last node found navigates forward
temp2 = temp2->next; // Move to next link in chain
}
temp2->next = temp; // new node (temp) appended at the end of the list
```

Question sources

Computer Science MCQ practice collection · Question 409

Question 152 | AM0410

Which data structure can be efficiently used for implementing undo and redo functionality in applications?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: Stack

Undo reverses the most recent action first, matching the source stack backtracking/reversal uses. A second stack can retain undone actions and reverse them again for redo. Evidence: assigned PDF pages 7, 6.

Why each choice is right or wrong

A - Correct: Push records actions; pop undoes the most recent; a second LIFO stack supports redo. See assigned PDF 7, 6.

B - Incorrect: FIFO would undo the oldest pending action first. See assigned PDF 7, 6.

C - Incorrect: A generic linked list can implement a stack but supplies no undo discipline by itself. See assigned PDF 7, 6.

D - Incorrect: A binary tree does not directly supply last-action-first reversal. See assigned PDF 7, 6.

Primary note proof

Study-note section DSA12

> Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

> • Reversing Data and Reverse a List
> • Convert Decimal to Binary
> • Parsing - It is a logic that breaks into independent pieces for further processing
> • Backtracking
> Examples:
> 1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
> 2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science MCQ practice collection · Question 410

Question 153 | AM0411

A forward-only input sequence must be emitted in reverse order after all input is read, using one auxiliary ADT through its standard insertion/removal interface. Which ADT directly reverses the order without modifying the input links or adding a separate reversal algorithm?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: A. Stack

A stack accepts the sequence in arrival order and then removes the most recently inserted item first. Repeated pops therefore emit the complete reverse sequence directly. A FIFO queue preserves the original order. A generic linked list or binary tree does not itself specify the required removal discipline; reversing existing links would be a separate algorithm excluded by this question. The contract identifies the auxiliary reversal ADT.

Why each choice is right or wrong

A - Correct: Push all items and then pop to emit reverse order directly.

B - Incorrect: FIFO removal preserves input order.

C - Incorrect: A generic list has no inherent reverse-removal contract without an added algorithm or stack policy.

D - Incorrect: A generic tree supplies no required reverse-sequence service.

Primary note proof

Study-note section DSA12

> Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

• Reversing Data and Reverse a List
• Convert Decimal to Binary
• Parsing - It is a logic that breaks into independent pieces for further processing
• Backtracking
Examples:
1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Data Structure and Algorithms module | PDF page 6 | lines 15-22

“A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 5 | lines 3-8

While adding a node to a list, the chain needs to be rearranged properly so that the each nodes are connected to each other. It is possible to add a node anywhere in the chain (at the start, middle or end). Let’s consider adding a node at the end of the list:
Step 1: First we need to reserve a space for the new node using the keyword new.
Node *temp = new *Node; // Reserving a space for temporary pointer which points to new node;

Question sources

Computer Science MCQ practice collection · Question 411

Question 154 | AM0412

Which abstract frontier service policy directly implements the latest-pending-work-first behavior of depth-first search? Choose the ADT, rather than a storage representation that could be given that policy.

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: A. Stack

DFS finishes the most recently entered unfinished branch before returning to older pending work. A stack directly supplies this latest-pending-work-first rule. A queue would instead remove older pending items first, producing a breadth-first discipline under ordinary graph traversal. A linked list may implement a stack, but its links alone do not prescribe the frontier policy. A binary tree describes possible input relationships rather than the required pending-work ADT.

Why each choice is right or wrong

A - Correct: A stack directly supplies latest-pending-work-first service.

B - Incorrect: A FIFO queue retrieves older pending work first.

C - Incorrect: Linked-list storage alone does not specify a frontier service policy.

D - Incorrect: A binary tree is not the latest-pending-work service ADT.

Primary note proof

Study-note section DSA42

> Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours. An explicit stack can store pending work instead. Both implement latest-pending-work first; a queue would change that discipline. Marking prevents infinite repetition around graph cycles.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

> • Reversing Data and Reverse a List
> • Convert Decimal to Binary
> • Parsing - It is a logic that breaks into independent pieces for further processing
> • Backtracking

Examples:
1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Question sources

Computer Science MCQ practice collection · Question 412

Question 155 | AM0413

Which listed representations can implement a priority queue without retaining an unmodified LIFO or FIFO removal contract? Interpret D as a binary heap and C as an ordered or unordered list using extremum selection. Select all that apply.

A. Stack

B. Queue

C. Linked list

D. Binary tree organized as a binary heap

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C, D: C. Linked list; D. Binary tree organized as a binary heap

A priority queue removes an item with an extreme priority key. An unordered list can scan for that extremum, while an ordered list maintains the relevant order during insertion. A binary heap supplies the same priority contract through heap order and logarithmic insertion/extraction. An unmodified stack or FIFO queue removes according to arrival position rather than priority. Thus the list and heap-shaped binary tree qualify under the stated algorithms.

Why each choice is right or wrong

A - Incorrect: An unmodified stack removes by recency rather than priority.

B - Incorrect: An unmodified FIFO queue removes by arrival order rather than priority.

C - Correct: Ordered or scanned lists can select the extreme priority.

D - Correct: A binary heap implements priority selection through heap order.

Primary note proof

Study-note section DSA17

A priority queue returns the smallest or largest priority key under its convention. Urgent patients may precede earlier arrivals. Insertions and extractions can interleave. Binary heaps support constant peek and logarithmic insert/extract.

Study-note section DSA17

Arrays and lists can also implement priority queues: an unordered sequence gives cheap insertion but linear extremum removal; an ordered sequence reverses that tradeoff. Equal priorities have no inherent FIFO guarantee. If stable service is required, compare (priority, arrival_counter). Choose the priority direction and tie rule before tracing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 40-45

Priority Queue:- is a queue where each data has an associated key provided at insertion time.
Application of Queue
Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
• Print server- maintains a queue of print jobs
• Disk Driver- maintains a queue of disk input/output requests
• Task scheduler in multiprocessing system- maintains priority queues of processes

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Question sources

Computer Science MCQ practice collection · Question 413

Question 156 | AM0416

Which ordinary ADT restricts insertion to the rear and removal or peek to the front, preserving arrival order? Assume standard ADT interfaces rather than additional traversal operations.

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B: B. Queue

The ordinary queue inserts new items at its rear and removes or observes the front item. These restrictions produce FIFO service: earlier arrivals are removed before later arrivals. A stack uses a single top and LIFO order. A general linked list permits operations beyond these restricted endpoints, while a binary tree organizes hierarchical relationships. Specifying the exact operations resolves the original vague use of access.

Why each choice is right or wrong

A - Incorrect: A stack operates at one top using LIFO.

B - Correct: A queue has rear insertion and front removal/peek.

C - Incorrect: A general list does not restrict access to those two operations.

D - Incorrect: A binary tree does not have this FIFO endpoint contract.

Primary note proof

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 8 | lines 38-40

> Deque (pronounced as Deck): is a Double Ended Queue in which insertion and deletion can occur at either end.
> Priority Queue:- is a queue where each data has an associated key provided at insertion time.

Question sources

Computer Science MCQ practice collection · Question 416

Question 157 | AM0417

Which structures can use nodes connected by pointers in a linked implementation? Do not require that every implementation use pointers. Select all that apply.

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A, B, C, D: A. Stack; B. Queue; C. Linked list; D. Binary tree

Pointer links can connect the nodes of several structures. A linked list uses next links, a linked stack operates at the chain head, and a linked queue maintains endpoint references around a chain. A pointer-based binary tree instead links child nodes along branches. All four therefore admit a node-and-pointer representation. Their alternative array layouts do not invalidate this existence claim about linked implementations.

Why each choice is right or wrong

A - Correct: A linked stack connects nodes through head-chain links.

B - Correct: A linked queue connects nodes and retains endpoint pointers.

C - Correct: A linked list directly uses node links.

D - Correct: A pointer-based tree links its child nodes.

Primary note proof

Study-note section DSA-S03

A linked binary-tree node stores a payload and two child pointers, one for the left child and one for the right child. A null child pointer means that child is absent; a root pointer gives access to the whole tree. In a dynamically allocated version, insertion can allocate a node and attach it through the appropriate child link.

Study-note section DSA01

Arrays or linked nodes can implement the same LIFO stack.

Study-note section DSA14

A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head.

Study-note section DSA08

A singly linked node contains data and next.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Question sources

Computer Science MCQ practice collection · Question 417

Question 158 | AM0419

Which data structure allows elements to be inserted and removed from both ends?

A. Stack

B. Queue

C. Deque

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: Deque

A double-ended queue permits insertion and deletion at either end by definition. Evidence: assigned PDF pages 8.

Why each choice is right or wrong

A - Incorrect: Stack restricts both operations to one top. See assigned PDF 8.

B - Incorrect: Ordinary queue fixes insertion rear and deletion front. See assigned PDF 8.

C - Correct: Deque is expressly defined by operations at both ends. See assigned PDF 8.

D - Incorrect: A binary tree is a hierarchy, not a double-ended sequence. See assigned PDF 8.

Primary note proof

Study-note section DSA16

> A deque permits both-end insertion/removal without guaranteeing middle access. Endpoint-linked operations are constant but middle indexing is linear. A circular-array deque may support indexed access. Dequeue names removal; deque names the structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 38-40

> Deque (pronounced as Deck): is a Double Ended Queue in which insertion and deletion can occur at either end.
> Priority Queue:- is a queue where each data has an associated key provided at insertion time.

Question sources

Computer Science MCQ practice collection · Question 419

Question 159 | AM0420

Which ADTs impose an item-removal order through their ordinary service contract? Use LIFO for stacks, FIFO for queues, and smallest-priority-first with FIFO ties for priority queues. A generic binary tree has no specified traversal. Select all that apply.

A. Stack

B. Queue

C. Priority queue

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A, B, C: A. Stack; B. Queue; C. Priority queue

Stacks impose reverse arrival order and queues impose arrival order. A priority queue also imposes an order once the priority direction and tie rule are supplied; this version chooses the smallest priority and serves equal priorities FIFO. A generic binary tree supplies a shape with several possible traversals rather than a single inherent removal order. Consequently the three service ADTs qualify and the unspecified tree does not.

Why each choice is right or wrong

A - Correct: A stack imposes LIFO removal.

B - Correct: A queue imposes FIFO removal.

C - Correct: The stated priority and tie rules define priority-queue removal order.

D - Incorrect: A generic tree without a traversal has no single inherent removal order.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA14

> This is FIFO or first-come, first-served.

Study-note section DSA17

> Arrays and lists can also implement priority queues: an unordered sequence gives cheap insertion but linear extremum removal; an ordered sequence reverses that tradeoff. Equal priorities have no inherent FIFO guarantee. If stable service is required, compare (priority, arrival_counter). Choose the priority direction and tie rule before tracing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 8 | lines 40-45

> Priority Queue:- is a queue where each data has an associated key provided at insertion time.
> Application of Queue
> Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
> • Print server- maintains a queue of print jobs
> • Disk Driver- maintains a queue of disk input/output requests
> • Task scheduler in multiprocessing system- maintains priority queues of processes

Data Structure and Algorithms module | PDF page 9 | lines 40-44

> III. Tree Traversing:
> Binary search tree can be traversed in three ways.
> a. Pre order traversal - traversing binary tree in the order of parent, left and right.
> b. Inorder traversal - traversing binary tree in the order of left, parent and right.
> c. Postorder traversal - traversing binary tree in the order of left, right and parent.

Question sources

Computer Science MCQ practice collection · Question 420

Question 160 | AM0421

Which is the most general listed binary-tree family defined by the following distinct-key invariant alone, without an additional balancing discipline: every key in a node's left subtree is smaller, and every key in its right subtree is larger?

A. AVL tree

B. Binary search tree

C. Red-black tree

D. B-tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B: B. Binary search tree

The global left-smaller/right-larger invariant defines the binary search tree family. It applies to every descendant in each subtree, not only immediate children. AVL and red-black trees satisfy that ordering but impose additional balancing disciplines, so they are more specialized than the family requested. The binary-tree restriction also excludes a general B-tree. Asking for the most general family prevents those BST subclasses from becoming competing selections.

Why each choice is right or wrong

A - Incorrect: AVL adds balance requirements beyond the stated invariant.

B - Correct: BST is the general family defined by the strict subtree ordering alone.

C - Incorrect: Red-black trees add a balancing discipline beyond the stated invariant.

D - Incorrect: A general B-tree is not the requested binary-tree family.

Primary note proof

Study-note section DSA19

> With distinct keys, every key in a node's left subtree is smaller and every key in its right subtree is larger. Immediate-child comparisons alone are insufficient: root 10, left child 5, and 5's right child 12 violate the root's restriction. Duplicates require a separate count or placement policy.

Study-note section DSA22

> AVL balance factor is height(left)-height(right); every node must have factor -1,0 or 1. After insertion or deletion, update heights upward and rotate where necessary. LL imbalance needs a right rotation; RR needs left; LR rotates the left child left then the node right; RL is symmetric. Rotations preserve inorder key order.

Study-note section DSA-S03

> A B-tree is a balanced multiway search tree. Its allowed branching factor can exceed two, so a general B-tree is not the binary-tree family. Balanced describes a height discipline; it does not by itself restrict every node to two children.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Question sources

Computer Science MCQ practice collection · Question 421

Question 161 | AM0422

Which data structure represents a hierarchical structure with a set of connected nodes?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer D: Binary tree

The module defines trees as hierarchical structures consisting of nodes and connecting edges. Binary trees specialize that hierarchy to two possible children. Evidence: assigned PDF pages 9.

Why each choice is right or wrong

A - Incorrect: Stack is a linear LIFO list. See assigned PDF 9, 9.

B - Incorrect: Queue is a linear FIFO list. See assigned PDF 9, 9.

C - Incorrect: Single linked list is a sequential chain. See assigned PDF 9, 9.

D - Correct: Binary tree is the hierarchical option. See assigned PDF 9, 9.

Primary note proof

Study-note section DSA18

A rooted tree has one parentless root and one parent per other node. It is connected and acyclic, has n-1 edges, and one path between any pair. Depth counts root-to-node edges; height is maximum depth. A single-node tree therefore has height zero. Leaves have no children.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 3-13

A tree is a set of nodes and edges that connect pairs of nodes that connect pairs of nodes. It is an abstract model of a hierarchical structure. Rooted tree has the following structure:
• One node distinguished as root.
• Every node C except the root is connected from exactly other node P. P is C's parent, and C is one of C's children.
• There is a unique path from the root to the each node.
• The number of edges in a path is the length of the path.
Tree Terminologies:
• Root: a node without a parent.
• Internal node: a node with at least one child.
• External (leaf) node: a node without a child.

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Question sources

Computer Science MCQ practice collection · Question 422

Question 162 | AM0423

Which data structure allows efficient insertion and deletion at both ends?

A. Stack

B. Queue

C. Deque

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: Deque

The source deque definition allows insertion and deletion at either end. With endpoint references these are local operations; no middle search is needed. Evidence: assigned PDF pages 8.

Why each choice is right or wrong

A - Incorrect: Stack offers only the top endpoint. See assigned PDF 8.

B - Incorrect: Queue fixes rear insertion and front deletion, not both operations at both ends. See assigned PDF 8.

C - Correct: Deque supplies the requested two-ended operations. See assigned PDF 8.

D - Incorrect: Binary trees do not define two list ends. See assigned PDF 8.

Primary note proof

Study-note section DSA16

> A deque permits both-end insertion/removal without guaranteeing middle access. Endpoint-linked operations are constant but middle indexing is linear. A circular-array deque may support indexed access. Dequeue names removal; deque names the structure.

Study-note section DSA10

> Doubly linked nodes add prev; head and tail enable constant-time operations at both ends. Singly linked tail deletion still needs its predecessor. A circular tail links back to head; stop traversal on returning to the start, handling empty separately.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 38-40

> Deque (pronounced as Deck): is a Double Ended Queue in which insertion and deletion can occur at either end.
> Priority Queue:- is a queue where each data has an associated key provided at insertion time.

Question sources

Computer Science MCQ practice collection · Question 423

Question 163 | AM0425

Which data structure uses the concept of "first in, last out" to handle function calls and recursion?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: Stack

FILO is explicitly equivalent to LIFO, and the source names stack memory for calls and recursion. The most recent call returns before the earlier waiting caller. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Correct: Stack supplies FILO nesting for function frames. See assigned PDF 6, 6.

B - Incorrect: Queue is FIFO and would not match nested returns. See assigned PDF 6, 6.

C - Incorrect: A linked list can store frames but does not itself impose call order. See assigned PDF 6, 6.

D - Incorrect: A binary tree does not implement the stated endpoint discipline by its definition. See assigned PDF 6, 6.

Primary note proof

Study-note section DSA12

> Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 6 | lines 46-52

> Applications of Stack
> • It is very useful to evaluate arithmetic expressions. (Postfix Expressions)
> • Infix to Postfix Transformation
> • It is useful during the execution of recursive programs

• A Stack is useful for designing the compiler in operating system to store local variables inside a function block.
• A stack (memory stack) can be used in function calls including recursion.

Question sources

Computer Science MCQ practice collection · Question 425

Question 164 | AM0427

Which data structure is suitable for implementing depth-first search (DFS) algorithm?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: Stack

DSA42 describes recursively finishing an unvisited neighbor before returning to the parent. Explicit stack storage or the runtime call stack preserves this latest-pending-work-first behavior. A FIFO queue changes the exploration order. Linked lists can store frames but do not themselves define stack order, and a binary tree alone is not a frontier policy.

Why each choice is right or wrong

A - Correct: A stack stores pending frames or vertices so the latest unfinished branch is resumed first.

B - Incorrect: A queue serves the oldest pending work and changes the depth-first exploration discipline.

C - Incorrect: A linked list may store a stack, but its name alone supplies no depth-first service order.

D - Incorrect: A binary tree is not the standard pending-work ADT needed to preserve DFS return points.

Primary note proof

Study-note section DSA42

Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours. An explicit stack can store pending work instead. Both implement latest-pending-work first; a queue would change that discipline. Marking prevents infinite repetition around graph cycles.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

• Reversing Data and Reverse a List
• Convert Decimal to Binary
• Parsing - It is a logic that breaks into independent pieces for further processing
• Backtracking
Examples:
1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Data Structure and Algorithms module | PDF page 6 | lines 15-22

“A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science MCQ practice collection · Question 427

Question 165 | AM0428

A capacity-limited cache evicts entries strictly in insertion order; cache hits do not change that order. Which ADT should maintain its eviction order? Ask for the service policy rather than a possible storage implementation.

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B: B. Queue

The earliest inserted resident entry must be evicted first, and hits do not move it. This is FIFO order, which a queue directly maintains through rear insertion and front removal. A linked list could be used to implement that queue but does not independently prescribe the eviction discipline. A stack reverses insertion order, and a generic binary tree provides no FIFO eviction contract.

Why each choice is right or wrong

A - Incorrect: LIFO would evict the most recent insertion rather than the oldest.

B - Correct: FIFO queue service matches insertion-order eviction.

C - Incorrect: A linked list is a possible representation but does not itself prescribe FIFO service.

D - Incorrect: A generic binary tree does not prescribe insertion-order eviction.

Primary note proof

Study-note section DSA27

> A dictionary or compiler symbol table can use hashing for expected constant lookup, or a balanced tree for logarithmic lookup with ordering. A bounded cache also needs eviction rules: FIFO uses a queue; LRU can combine hash lookup with a doubly linked recency list. Move a hit to the recent end and evict the least-recent end when full. No storage structure is universally best without operation and policy requirements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

> “A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
> The two basic operations are:
> Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
> Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various

operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Question sources

Computer Science MCQ practice collection · Question 428

Question 166 | AM0429

Which data structure is commonly used for implementing undo-redo functionality in text editors?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: Stack

Undo applies the inverse of the latest action, a LIFO operation. Keeping undone actions on another stack supports redo in the reverse order of undo. Evidence: assigned PDF pages 7, 6.

Why each choice is right or wrong

A - Correct: Two stacks implement the required action reversal and restoration ordering. See assigned PDF 7, 6.

B - Incorrect: FIFO would undo actions in their original order. See assigned PDF 7, 6.

C - Incorrect: A linked list can store history but needs an added stack discipline. See assigned PDF 7, 6.

D - Incorrect: A binary tree does not intrinsically select the latest action. See assigned PDF 7, 6.

Primary note proof

Study-note section DSA12

Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 2-8

• Reversing Data and Reverse a List
• Convert Decimal to Binary
• Parsing - It is a logic that breaks into independent pieces for further processing
• Backtracking
Examples:
1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Data Structure and Algorithms module | PDF page 6 | lines 15-22

“A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science MCQ practice collection · Question 429

Question 167 | AM0430

Which data structure allows easy traversal in a Last-In-First-Out (LIFO) order?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: Stack

Easy traversal here is interpreted as repeated access/removal by the defining ADT rule. A stack pops the most recently pushed remaining element first. Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Correct: Stack expressly enforces last-in-first-out. See assigned PDF 6.

B - Incorrect: Queue expressly enforces first-in-first-out. See assigned PDF 6.

C - Incorrect: A linked list can emulate LIFO only after choosing an endpoint policy. See assigned PDF 6.

D - Incorrect: A binary tree traversal is determined by child order, not insertion recency. See assigned PDF 6.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

> “A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
> • Push To add an element to the stack
> • Pop To remove an element from the stock
> • Peek To look at elements in the stack without removingthem
> • LIFO Refers to the last in, first out behavior of thestack
> • FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science MCQ practice collection · Question 430

Question 168 | AM0431

Which specified representation supports constant-time insertion/deletion at either endpoint and constant-time indexed access to the logical middle item? Assume fixed capacity, no shifting or resizing, spare space for insertion, and a nonempty sequence for access.

A. Stack

B. Queue

C. Fixed-capacity circular-array deque

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: C. Fixed-capacity circular-array deque

A circular-array deque maps logical position k to a slot by adding the front index and reducing modulo capacity. That makes the middle item directly indexable. Endpoint insertions and deletions change only a slot and constant-size indices when no resize or shift occurs. A linked deque would not provide this indexing guarantee, so the array representation is explicit. The other ordinary ADT interfaces lack the combined operation set.

Why each choice is right or wrong

A - Incorrect: A stack does not expose both-end updates and middle indexing.

B - Incorrect: An ordinary queue does not expose all four endpoint operations or arbitrary middle access.

C - Correct: The fixed circular-array deque supplies endpoint updates and indexed middle access.

D - Incorrect: A generic binary tree lacks constant-time access by middle sequence position.

Primary note proof

Study-note section DSA16

> A deque permits both-end insertion/removal without guaranteeing middle access. Endpoint-linked operations are constant but middle indexing is linear. A circular-array deque may support indexed access. Dequeue names removal; deque names the structure.

Study-note section DSA15

> insertion slot is (front+size)%C.

Study-note section DSA02

> This arithmetic gives constant-time positional access in the usual RAM cost model.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 8 | lines 38-40

> Deque (pronounced as Deck): is a Double Ended Queue in which insertion and deletion can occur at either end.
> Priority Queue:- is a queue where each data has an associated key provided at insertion time.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node

pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

“A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists. The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to the user. An underflow error occurs if the queue is empty.

Question sources

Computer Science MCQ practice collection · Question 431

Question 169 | AM0432

Which data structure is commonly used for implementing the undo operation in a text editor?

A. Stack

B. Queue

C. Linked list

D. Binary tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: Stack

The latest action must be reversed first, matching LIFO and the source backtracking application. Pushing action records and popping the latest implements that order. Evidence: assigned PDF pages 6, 7.

Why each choice is right or wrong

A - Correct: Top holds the latest pending action to undo. See assigned PDF 6, 7.

B - Incorrect: FIFO selects the oldest pending action instead. See assigned PDF 6, 7.

C - Incorrect: A generic linked list needs a chosen stack discipline to implement undo. See assigned PDF 6, 7.

D - Incorrect: A binary tree alone does not impose reverse chronology. See assigned PDF 6, 7.

Primary note proof

Study-note section DSA12

Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

“A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Data Structure and Algorithms module | PDF page 7 | lines 2-8

• Reversing Data and Reverse a List
• Convert Decimal to Binary
• Parsing - It is a logic that breaks into independent pieces for further processing
• Backtracking
Examples:
1. Infix notation: A+(B*C) Equivalent Postfix notation ABC*+
2. Infix notation: (A+B)*C Equivalent Postfix notation AB+C*

Question sources

Computer Science MCQ practice collection · Question 432

Question 170 | AM0433

A bounded FIFO buffer must reuse freed array slots by wrapping indices modulo its fixed capacity, without shifting items. New insertions are rejected while full. Which listed term specifically names this storage representation?

A. Stack

B. Queue

C. Linked list

D. Circular buffer

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer D: D. Circular buffer

A circular buffer reuses its fixed slots by wrapping indices modulo capacity, so released positions can be used again without moving surviving items. The FIFO order is tracked logically rather than by requiring the front to remain at physical slot zero. Queue names the service ADT and can have other representations; it does not specifically identify this layout. Stack and linked list likewise do not name the stated circular-array representation.

Why each choice is right or wrong

A - Incorrect: A stack names LIFO service rather than the specified FIFO ring storage.

B - Incorrect: Queue names the FIFO ADT but does not specifically identify its storage layout.

C - Incorrect: A linked list does not name modulo-indexed circular array slots.

D - Correct: Circular buffer specifically names the stated fixed-slot wraparound representation.

Primary note proof

Study-note section DSA15

> A simple noncircular array queue may report overflow when rear reaches the final index despite released slots at the front. A circular queue reuses slots with modulo arithmetic. Use a count convention: front points to the next removal, size counts items, and insertion slot is (front+size)%C. Empty means size==0; full means size==C.

Study-note section DSA16

> Merging FIFO queues needs an explicit policy: append one sequence after another or alternate their fronts. Splitting by a predicate preserves each subgroup’s original order if each removed item is appended to its matching output queue. A fixed-capacity buffer also needs a full-policy choice: reject, block or overwrite. Capacity alone does not choose a policy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 7 | lines 24-26

> II. Circular array implementation of Queue Operations
> To overcome the above drawback of simple array implementation a circular array can be used, in
> which freed spaces are re-used to store data.

Data Structure and Algorithms module | PDF page 7 | lines 18-23

> I. Simple Array implementation (static memory).
> A simple array structure can be used to implement enqueue & dequeue operations. Thus item will
> be added at the end of the array whereas dequeue operation removes items from the front. In

simple
array implementation, to add item it checks whether there is room at the rear or not. Therefore, even though there are free released spaces at the front ot the array the queue implementation might
generate overflow message. This is the key drawback to use simple array to implement Queue.

Question sources

Computer Science MCQ practice collection · Question 433

Question 171 | AM0434

Which ordinary array sorting algorithms have tight worst-case time Theta(n^2)? Select all that apply. Use buffered mergesort, ordinary quicksort, standard insertion sort and iterative heapsort.

A. Merge sort

B. Quick sort

C. Insertion sort

D. Heap sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B, C: B. Quick sort; C. Insertion sort

Repeated extreme partitions make ordinary quicksort accumulate quadratic work. Insertion sort on reverse input similarly accumulates quadratic prefix shifts. Buffered mergesort and iterative heapsort have n log n worst-case bounds instead, so they are not tightly quadratic even though quadratic is a valid looser upper bound. The explicit tight-bound question therefore selects quicksort and insertion alone in this primary option set.

Why each choice is right or wrong

A - Incorrect: Buffered mergesort has n log n worst growth.

B - Correct: Extreme quicksort partitions give tight quadratic worst growth.

C - Correct: Reverse-order insertion gives tight quadratic worst growth.

D - Incorrect: Iterative heapsort has an n log n worst bound.

Primary note proof

Study-note section DSA36

Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work.

Study-note section DSA34

Reverse order causes 1+2+...+(n-1) shifts: quadratic time.

Study-note section DSA37

There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA38

Building with bottom-up sinking is linear; repeated extraction gives O(n log n) worst time.

Related reading (context, not primary note proof)

Study-note section DSA32

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Variant 1 | Alternative version 1

Which ordinary array sorting algorithms have tight worst-case time Theta(n^2)? Use strict-swap bubble sort, ordinary quicksort, buffered mergesort and standard insertion sort. Select all that apply.

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Answer A, B, D: A. Bubble sort; B. Quick sort; D. Insertion sort

Bubble sort can require quadratically many adjacent comparisons, quicksort can accumulate quadratic work through extreme pivots, and insertion sort can perform quadratic shifts on reversed input. Buffered mergesort has n log n growth instead. The variant has bubble, quick, merge and insertion in its own label order, so its complete tight-quadratic selection is A, B and D.

Why each choice is right or wrong

A - Correct: Adjacent bubble passes can require quadratic work.

B - Correct: Extreme quicksort partitions can require quadratic work.

C - Incorrect: Buffered mergesort has n log n growth.

D - Correct: Reverse input causes quadratic insertion shifts.

Primary note proof

Study-note section DSA36

> Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work.

Study-note section DSA34

> Reverse order causes 1+2+...+(n-1) shifts: quadratic time.

Study-note section DSA32

> Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Related reading (context, not primary note proof)

Study-note section DSA38

Question sources

Computer Science MCQ practice collection · Question 434

Computer Science MCQ practice collection · Question 444

Question 172 | AM0435

Which sorting algorithm uses the divide-and-conquer strategy? Use ordinary array implementations, with radix sorting specifically meaning iterative least-significant-digit passes.

A. Bubble sort

B. Selection sort

C. Merge sort

D. Radix sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: C. Merge sort

Mergesort divides the array into two halves, recursively sorts them and merges their results, so it uses divide and conquer. Ordinary bubble sort uses adjacent passes and selection sort repeatedly selects a suffix minimum. The explicitly specified LSD radix method performs stable digit passes rather than recursive subproblem decomposition. This convention removes the possible confusion with MSD radix grouping while retaining the original primary answer.

Why each choice is right or wrong

A - Incorrect: Ordinary bubble sort uses adjacent passes rather than recursive decomposition.

B - Incorrect: Ordinary selection sort repeatedly places one suffix minimum.

C - Correct: Mergesort recursively solves smaller halves and combines their results.

D - Incorrect: The specified iterative LSD form uses digit passes rather than recursive divide and conquer.

Primary note proof

Study-note section DSA-S04

> Divide and conquer reduces a problem to smaller instances of the same kind, solves the required smaller instances and obtains the original result from them. Mergesort recursively sorts two halves and merges; quicksort partitions and recursively sorts the resulting ranges. This course also classifies binary search as divide and conquer: an ordered comparison selects the only half that can contain the target. This is different from simply repeating adjacent swaps, choosing successive minima, or making stable LSD digit passes.

Study-note section DSA39

> LSD radix sorting handles fixed-width digit keys by stably sorting least-significant digit first, then moving left. Stable intermediate passes preserve the ordering already established by less-significant digits. With d digits and radix r, counting-based passes take O(d(n+r)) time and O(n+r) auxiliary storage.

Study-note section DSA34

> Begin with the first element already sorted. For each remaining item, save its value, shift larger preceding values right, and insert it into the gap.

Related reading (context, not primary note proof)

Study-note section DSA36

Study-note section DSA37

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 38-44

> iv. Merge sort
> Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
> Algorithm:

1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Variant 1 | Alternative version 1

Which ordinary sorting algorithms use divide and conquer by recursively sorting smaller subproblems? Select all that apply.

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Answer B, C: B. Quick sort; C. Merge sort

Quicksort partitions around a pivot and recursively sorts the smaller ranges. Top-down mergesort recursively sorts halves and merges their results. Both therefore apply divide and conquer. Bubble sort's adjacent passes and ordinary insertion sort's growing sorted prefix do not use this recursive subproblem decomposition. The complete answer is B and C; the question requires multiple selection.

Why each choice is right or wrong

A - Incorrect: Bubble sorting repeatedly swaps adjacent inversions instead of recursively decomposing subinstances.

B - Correct: Quicksort partitions around a pivot and recursively sorts the smaller ranges.

C - Correct: Merge sort recursively sorts halves and combines their sorted results.

D - Incorrect: Ordinary insertion sort extends a sorted prefix instead of applying recursive divide and conquer.

Primary note proof

Study-note section DSA-S04

> Divide and conquer reduces a problem to smaller instances of the same kind, solves the required smaller instances and obtains the original result from them. Mergesort recursively sorts two halves and merges; quicksort partitions and recursively sorts the resulting ranges. This course also classifies binary search as divide and conquer: an ordered comparison selects the only half that can contain the target. This is different from simply repeating adjacent swaps, choosing successive minima, or making stable LSD digit passes.

Study-note section DSA36

> Choose a pivot, partition, then recursively sort the pieces. Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work. Expected n log n needs an input/random-pivot model. Random and median-of-three choices have different tradeoffs; neither is universally best.

Study-note section DSA37

> Split into halves, sort them, then merge by advancing the smaller current front. With left-first tie handling, equal keys retain order. Ordinary array merging needs a linear auxiliary buffer. There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA34

> Begin with the first element already sorted. For each remaining item, save its value, shift larger preceding values right, and insert it into the gap.

Related reading (context, not primary note proof)

Study-note section DSA39

Question sources

Computer Science MCQ practice collection · Question 435

Computer Science MCQ practice collection · Question 445

Question 173 | AM0436

Which specified sorting implementations preserve the relative order of equal full keys? Use ordinary in-place quicksort, ordinary heapsort, LSD radix with stable digit passes, and mergesort with left-first ties. Select all that apply.

A. Quick sort

B. Heap sort

C. Radix sort

D. Merge sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C, D: C. Radix sort; D. Merge sort

Stable LSD radix passes preserve the ordering already established by less significant digits. Equal full keys remain tied on every pass, so their relative order remains unchanged. Mergesort with left-first ties also preserves equal-key order while combining runs. Ordinary in-place quicksort and heapsort do not guarantee that stability because distant exchanges can reorder equal records. Hence both radix and merge qualify under the specified implementations.

Why each choice is right or wrong

A - Incorrect: Ordinary quicksort does not guarantee equal-key stability.

B - Incorrect: Long heap exchanges can reorder equal records.

C - Correct: Stable digit passes preserve the order of equal full keys.

D - Correct: Left-first merge ties preserve equal-key order.

Primary note proof

Study-note section DSA39

> Stable intermediate passes preserve the ordering already established by less-significant digits.

Study-note section DSA37

> With left-first tie handling, equal keys retain order.

Study-note section DSA38

> The method is generally unstable because long root-to-end exchanges can reorder equal keys.

Study-note section DSA31

stack usually log n average, n worst; unstable

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Computer Science MCQ practice collection · Question 436

Question 174 | AM0437

Which sorting algorithm uses a pivot element to partition the array into two subarrays?

A. Bubble sort

B. Selection sort

C. Quick sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: Quick sort

The module quicksort explicitly selects a pivot and partitions values below/above it before recursion. Evidence: assigned PDF pages 11, 4, 3.

Why each choice is right or wrong

A - Incorrect: Bubble uses adjacent comparisons and swaps. See assigned PDF 11, 4, 3, 3.

B - Incorrect: Selection chooses the minimum for the next output slot. See assigned PDF 11, 4, 3, 3.

C - Correct: Pivot partitioning is the stated quicksort procedure. See assigned PDF 11, 4, 3, 3.

D - Incorrect: Insertion puts each item into a sorted prefix. See assigned PDF 11, 4, 3, 3.

Primary note proof

Study-note section DSA36

> Choose a pivot, partition, then recursively sort the pieces. Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work. Expected n log n needs an input/random-pivot model. Random and median-of-three choices have different tradeoffs; neither is universally best.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

> ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
> Algorithm:
> 1. Choose a pivot value (mostly the first element is taken as the pivot value)
> 2. Position the pivot element and partition the list so that:
> • the left part has items less than or equal to the pivot value
> • the right part has items greater than or equal to the pivot value
> 3. Recursively sort the left part
> 4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Computer Science MCQ practice collection · Question 437

Question 175 | AM0439

Which searching algorithm is based on comparing the target value with elements at midpoints of the array?

A. Linear search

B. Binary search

C. Interpolation search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B: Binary search

DSA29 computes the midpoint of the current ordered interval, compares it with the target, and discards the half that cannot contain the target. That is exactly the midpoint-based procedure in the stem. Linear search scans successive elements, interpolation estimates a position from values, and DFS follows graph branches rather than ordered-array midpoints.

Why each choice is right or wrong

A - Incorrect: Linear search compares successive elements rather than repeatedly choosing the midpoint of a shrinking ordered interval.

B - Correct: Binary search explicitly compares the target with the current interval midpoint and keeps one half.

C - Incorrect: Interpolation search estimates a probe from numeric values and their distribution, rather than always using the midpoint.

D - Incorrect: Depth-first search explores graph neighbors and branches, not ordered-array midpoint intervals.

Primary note proof

Study-note section DSA29

> On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Study-note section DSA30

> Interpolation search estimates the index from numeric values: lo+(hi-lo)×(key-A[lo])/(A[hi]-A[lo]), rounded to a valid index. Guard equal endpoints and out-of-range targets. On sorted, suitably uniform data its expected probes can be O(log log n); pathological distributions can require linear time. Binary search gives a logarithmic worst bound without that distribution assumption.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

> Linear/Sequential searching
> • Works with unsorted List
> • Loop through the list starting at the first
> element
> • Time is proportional to the size of input
> (n) and we call this time complexity
> O(n).

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Question sources

Computer Science MCQ practice collection · Question 439

Question 176 | AM0440

Which ordinary array-search algorithms require sorted input? For interpolation search also assume comparable numeric values and appropriate endpoint guards. Select all that apply.

A. Linear search

B. Binary search

C. Interpolation search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B, C: B. Binary search; C. Interpolation search

Binary search relies on order to discard half the remaining candidate interval after a comparison. Interpolation search also relies on sorted numeric values when estimating a candidate position from endpoint values. Both therefore require sorted input, although their further assumptions differ. Linear search simply scans and permits unsorted data. DFS follows graph adjacency rather than relying on a sorted array-search interval.

Why each choice is right or wrong

A - Incorrect: Linear scanning permits unsorted data.

B - Correct: Binary interval elimination needs sorted order.

C - Correct: Interpolation index estimation also needs sorted input.

D - Incorrect: DFS follows graph connections and has no sorted-array requirement.

Primary note proof

Study-note section DSA29

On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Study-note section DSA30

Interpolation search estimates the index from numeric values: lo+(hi-lo)×(key-A[lo])/(A[hi]-A[lo]), rounded to a valid index. Guard equal endpoints and out-of-range targets. On sorted, suitably uniform data its expected probes can be O(log log n); pathological distributions can require linear time. Binary search gives a logarithmic worst bound without that distribution assumption.

Study-note section DSA28

Unsorted data is allowed; sorted data is allowed too.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Computer Science MCQ practice collection · Question 440

Question 177 | AM0441

Which searching algorithm works by comparing the target value with elements from the beginning of the array until a match is found?

A. Linear search

B. Binary search

C. Interpolation search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: A. Linear search

Linear search starts at the beginning and compares successive elements until a match or exhaustion. Binary search instead chooses a midpoint of an ordered interval, and interpolation search estimates a position from endpoint key values. Depth-first search follows an unvisited graph branch before returning. These positive procedural differences classify the alternatives without any appeal to an old module's omissions, so A remains the definite matching choice.

Why each choice is right or wrong

A - Correct: Linear search compares successive elements from the beginning.

B - Incorrect: Binary search probes a midpoint rather than scanning every successive position.

C - Incorrect: Interpolation search estimates a probe position from endpoint values.

D - Incorrect: DFS follows graph adjacency and unfinished branches rather than a beginning-to-end array scan.

Primary note proof

Study-note section DSA28

> Compare the target with successive elements until equal or exhausted.

Study-note section DSA29

> Compute mid=lo+(hi-lo)//2.

Study-note section DSA30

> Interpolation search estimates the index from numeric values: lo+(hi-lo)×(key-A[lo])/(A[hi]-A[lo]), rounded to a valid index.

Study-note section DSA42

> Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Question sources

Computer Science MCQ practice collection · Question 441

Question 178 | AM0442

Which searching algorithm is more efficient for uniformly distributed data in a sorted array?

A. Linear search

B. Binary search

C. Interpolation search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: Interpolation search under the stated uniform-data model.

DSA30 gives interpolation search expected O(log log n) probes on sorted, suitably uniform numeric data. Binary search has logarithmic worst-case probes without that distribution assumption, while linear search scans. The stem supplies the distribution and sortedness conditions that motivate interpolation. The expected advantage does not promise faster finite runs or logarithmic worst-case behavior on arbitrary distributions.

Why each choice is right or wrong

A - Incorrect: Linear search still examines successive elements and does not exploit uniform numeric distribution for sublinear expected probing.

B - Incorrect: Binary search gives logarithmic worst-case probing, but does not exploit uniform values for the stated expected advantage.

C - Correct: Interpolation estimates positions from uniformly distributed sorted numeric values, supporting the note's expected O(log log n) probes.

D - Incorrect: DFS explores graph or tree branches and is not the distribution-based ordered-array search described here.

Primary note proof

Study-note section DSA30

> Interpolation search estimates the index from numeric values: lo+(hi-lo)×(key-A[lo])/(A[hi]-A[lo]), rounded to a valid index. Guard equal endpoints and out-of-range targets. On sorted, suitably uniform data its expected probes can be O(log log n); pathological distributions can require linear time. Binary search gives a logarithmic worst bound without that distribution assumption.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

> Linear/Sequential searching
> • Works with unsorted List
> • Loop through the list starting at the first element
> • Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Computer Science MCQ practice collection · Question 442

Question 179 | AM0443

Which named algorithms explore a graph or tree by following adjacency links to unvisited vertices, using branch-first or distance-layer-first traversal? Select all that apply.

A. Linear search

B. Binary search

C. Depth-first search

D. Breadth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C, D: C. Depth-first search; D. Breadth-first search

DFS explores an unvisited branch before returning to later neighbors, using recursion or a stack. BFS instead processes a FIFO frontier so successive distance layers are explored in order. Both follow adjacency relationships and apply to trees as well as general graphs. Linear sequence scanning and binary interval search are different algorithms and do not implement these two adjacency-frontier traversal disciplines.

Why each choice is right or wrong

A - Incorrect: Linear sequence search does not implement the specified adjacency-frontier traversal.

B - Incorrect: Binary search eliminates ordered interval halves rather than following arbitrary adjacency.

C - Correct: DFS is branch-first adjacency traversal.

D - Correct: BFS is distance-layer-first adjacency traversal.

Primary note proof

Study-note section DSA41

> Mark the source when enqueuing it. Repeatedly dequeue a vertex, examine its neighbours, and enqueue each unvisited neighbour while marking it. Store predecessor[v] when discovering v if a path must be reconstructed. FIFO ensures vertices at distance d are explored before later distance layers.

Study-note section DSA42

> Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours. An explicit stack can store pending work instead. Both implement latest-pending-work first; a queue would change that discipline. Marking prevents infinite repetition around graph cycles.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 40-44

III. Tree Traversing:
Binary search tree can be traversed in three ways.
a. Pre order traversal - traversing binary tree in the order of parent, left and right.
b. Inorder traversal - traversing binary tree in the order of left, parent and right.
c. Postorder traversal - traversing binary tree in the order of left, right and parent.

Question sources

Computer Science MCQ practice collection · Question 443

Question 180 | AM0446

Under a uniform random permutation of distinct keys, which ordinary sorting algorithms have tight average time Theta(n log n)? Use fixed-position-pivot quicksort, buffered mergesort without merge-skipping, and standard bubble/insertion implementations. Select all that apply.

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B, C: B. Quick sort; C. Merge sort

The random-order model supplies the average-case assumption needed for ordinary quicksort's expected n log n work. Buffered mergesort performs n log n work across input orders, so its average has the same order. Bubble and insertion have quadratic ordinary average behavior under this random permutation model. Tight bounds and the stated distribution make quicksort and mergesort the complete selection, rather than leaving average cost unspecified. The DSA31 comparison table supplies the average-case entries used in this comparison.

Why each choice is right or wrong

A - Incorrect: Ordinary bubble sorting has quadratic random-order average work.

B - Correct: The stated random-input model gives quicksort expected n log n work.

C - Correct: Buffered mergesort has n log n work in every input order.

D - Incorrect: Ordinary insertion has quadratic random-order average work.

Primary note proof

Study-note section DSA31

n log n / n log n / n^2

Study-note section DSA31

n / n^2 / n^2

Study-note section DSA37

There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd

item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Computer Science MCQ practice collection · Question 446

Question 181 | AM0447

Which sorting algorithm works by repeatedly selecting the smallest element from the unsorted portion and placing it at the beginning?

A. Bubble sort

B. Quick sort

C. Selection sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: Selection sort

The source selection sort scans for the smallest remaining element and swaps it with the current first unsorted position. Evidence: assigned PDF pages 3, 4, 11.

Why each choice is right or wrong

A - Incorrect: Bubble pushes elements through adjacent swaps. See assigned PDF 3, 4, 11, 3.

B - Incorrect: Quicksort partitions by a pivot, not necessarily the minimum. See assigned PDF 3, 4, 11, 3.

C - Correct: Exactly the minimum-selection and placement procedure given. See assigned PDF 3, 4, 11, 3.

D - Incorrect: Insertion places the next encountered element into a sorted prefix. See assigned PDF 3, 4, 11, 3.

Primary note proof

Study-note section DSA33

At position i, scan the remaining suffix for its minimum, then exchange it with A[i]. There are n-1 meaningful placements and n(n-1)/2 comparisons regardless of initial order. At most n-1

exchanges of distinct positions are needed. A loop that also swaps the last item with itself can count n swap calls.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort
Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items.
• How many comparisons? => O(n2
) and How many swaps? => O(n)

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in
the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Computer Science MCQ practice collection · Question 447

Question 182 | AM0448

Which ordinary array sorting algorithms are comparison sorts that use constant extra element storage inside the original array? Count recursive call stacks separately from element storage; use buffered mergesort for C. Select all that apply.

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A, B, D: A. Bubble sort; B. Quick sort; D. Insertion sort

Bubble sort swaps adjacent keys and insertion sort shifts a prefix while retaining one saved key. Quicksort partitions within the same array using constant extra element storage. Its recursive frames are counted separately under the explicitly stated convention. Buffered mergesort instead allocates a linear merge buffer. Therefore bubble, quick and insertion meet the element-storage definition, while a total-auxiliary-space question would need to account for quicksort's stack as well.

Why each choice is right or wrong

A - Correct: Bubble compares and swaps within the original array.

B - Correct: Quicksort partitions in the array; its recursive stack is separately counted.

C - Incorrect: Buffered mergesort needs linear extra element storage.

D - Correct: Insertion compares and shifts within the original array with one saved key.

Primary note proof

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right.

Study-note section DSA34

> For each remaining item, save its value, shift larger preceding values right, and insert it into the gap.

Study-note section DSA36

> Quicksort normally rearranges in the same array but recursive storage can be linear in the worst case.

Study-note section DSA37

Ordinary array merging needs a linear auxiliary buffer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.

• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Computer Science MCQ practice collection · Question 448

Question 183 | AM0449

Which searching algorithm requires the input array to be sorted?

A. Linear search

B. Binary search

C. Hash search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B: Binary search

DSA29 defines binary search over an ascending random-access sequence and reverses comparisons for descending data. Sortedness makes the half-elimination step sound. Linear scanning works without order; hashing needs a built table rather than sorted positions; DFS follows adjacency. Only Binary search among this option set requires a sorted input sequence for its ordinary operation.

Why each choice is right or wrong

A - Incorrect: Linear search explicitly permits unsorted data and compares items one after another.

B - Correct: Binary search relies on ordered values to determine which half can still contain the target.

C - Incorrect: Hash lookup needs a hash-table organization, not values arranged in sorted-array order.

D - Incorrect: Depth-first search follows graph or tree adjacency and does not require a sorted input array.

Primary note proof

Study-note section DSA29

On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi.

Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Study-note section DSA25

Direct addressing reserves a position for every possible key in a finite universe U, costing Theta(|U|) slots. A hash table uses h(key) in 0..m-1 and stores only actual entries plus bucket capacity. With h(k)=k mod 6, key 36 has home address 0; collisions may move its final storage position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Computer Science MCQ practice collection · Question 449

Question 184 | AM0450

Which searching algorithm has a worst-case time complexity of O(log n)?

A. Linear search

B. Binary search

C. Hash search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B: Binary search has logarithmic worst-case time.

DSA29 bounds binary-search probes by floor($\log_2 n$)+1 on nonempty random-access input. This supplies logarithmic worst-case time under constant-cost comparisons. Linear search may scan all items; adverse hash collisions may require linear work; DFS can inspect all vertices and edges. Their ordinary worst cases do not share binary search's logarithmic bound.

Why each choice is right or wrong

A - Incorrect: Linear search can inspect every item when the target is absent or last, giving linear worst-case time.

B - Correct: Each comparison discards about half the ordered interval, giving logarithmic worst-case probes.

C - Incorrect: Ordinary hash lookup can degrade to linear time under adverse collisions, despite expected constant-time performance.

D - Incorrect: DFS may visit all graph vertices and edges, rather than guarantee logarithmic worst-case work.

Primary note proof

Study-note section DSA29

> Worst probes are floor($\log_2 n$)+1 for n>0: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Study-note section DSA28

> Compare the target with successive elements until equal or exhausted. Unsorted data is allowed; sorted data is allowed too. A first-element match costs Theta(1). An absent target or last-position match needs n comparisons, giving Theta(n) worst time. Uniformly likely successful positions average (n+1)/2 comparisons, still linear.

Study-note section DSA26

> Separate chaining stores colliding entries in a bucket list. Load alpha=n/m is average chain length; expected operations are O(1+alpha) under suitable distribution assumptions, with O(n) worst cases. Open addressing keeps entries inside the array and probes until a key or an available slot is found.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Computer Science MCQ practice collection · Question 450

Question 185 | AM0451

Which search compares a target with consecutive sequence elements at a fixed stride of one, stopping at the first match or after exhausting the sequence? No hashing or other preprocessing is used.

A. Linear search

B. Binary search

C. Hash search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: A. Linear search

The stated unit stride means the algorithm inspects successive elements in order. It stops early when a comparison matches or reports failure after the sequence is exhausted, which is linear search. Binary search uses a shrinking midpoint interval rather than consecutive positions. Hash search starts from computed addresses, and DFS follows graph adjacency. Specifying stride one distinguishes ordinary scanning from larger-step jump-search interpretations.

Why each choice is right or wrong

A - Correct: Linear search compares successive elements until a match or exhaustion.

B - Incorrect: Binary search chooses midpoints rather than every consecutive element.

C - Incorrect: Hash search uses computed addresses rather than the stated scan.

D - Incorrect: DFS follows graph adjacency rather than consecutive sequence positions.

Primary note proof

Study-note section DSA28

> Compare the target with successive elements until equal or exhausted. Unsorted data is allowed; sorted data is allowed too. A first-element match costs Theta(1). An absent target or last-position match needs n comparisons, giving Theta(n) worst time. Uniformly likely successful positions average (n+1)/2 comparisons, still linear.

Study-note section DSA25

> Direct addressing reserves a position for every possible key in a finite universe U, costing Theta(|U|) slots. A hash table uses h(key) in 0..m-1 and stores only actual entries plus bucket capacity. With h(k)=k mod 6, key 36 has home address 0; collisions may move its final storage position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

> Linear/Sequential searching
> • Works with unsorted List
> • Loop through the list starting at the first element
> • Time is proportional to the size of input (n) and we call this time complexity O(n).

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
> • The computational time for this algorithm is proportional to log2 n.

Question sources

Computer Science MCQ practice collection · Question 451

Question 186 | AM0452

Which searching algorithm is suitable for unsorted arrays or lists?

A. Linear search

B. Binary search

C. Hash search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A: Linear search directly searches an unsorted sequence.

DSA28 explicitly allows unsorted data for linear search: compare successive elements until the target is found or the sequence ends. Binary search requires sortedness. Hash lookup can be useful after building a table, but it does not directly search the supplied ordinary list without that preprocessing. DFS presumes graph or tree adjacency rather than a simple sequence query.

Why each choice is right or wrong

A - Correct: Linear search can inspect an unsorted array or list directly without sorting or building another structure.

B - Incorrect: Binary search needs ordering to safely eliminate half the remaining candidate positions.

C - Incorrect: Hash search needs a constructed hash table; the stem supplies an ordinary unsorted array or list.

D - Incorrect: DFS explores graph or tree adjacency and is not the direct unsorted-sequence scan described here.

Primary note proof

Study-note section DSA28

> Compare the target with successive elements until equal or exhausted. Unsorted data is allowed; sorted data is allowed too. A first-element match costs Theta(1). An absent target or last-position match needs n comparisons, giving Theta(n) worst time. Uniformly likely successful positions average (n+1)/2 comparisons, still linear.

Study-note section DSA29

> On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Study-note section DSA25

Direct addressing reserves a position for every possible key in a finite universe U, costing Theta(|U|) slots. A hash table uses h(key) in 0..m-1 and stores only actual entries plus bucket capacity. With h(k)=k mod 6, key 36 has home address 0; collisions may move its final storage position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Question sources

Computer Science MCQ practice collection · Question 452

Question 187 | AM0453

Which searching algorithm is commonly used for traversing graphs or trees?

A. Linear search

B. Binary search

C. Hash search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer D: Depth-first search

DSA42 describes marking a vertex and recursively visiting each unmarked neighbor before resuming later neighbors. That procedure is standard depth-first traversal and applies to graphs and trees. Linear search scans a sequence, binary search uses an ordered interval, and hash search queries table addresses. None of those alternatives names the adjacency-following traversal in this option set.

Why each choice is right or wrong

A - Incorrect: Linear search scans sequence elements and is not the named adjacency-following graph/tree traversal.

B - Incorrect: Binary search reduces an ordered interval rather than visiting arbitrary graph or tree connections.

C - Incorrect: Hash search probes key-derived table addresses rather than traversing graph or tree edges.

D - Correct: DFS follows unvisited neighbors recursively or through an explicit stack, providing a standard traversal.

Primary note proof

Study-note section DSA42

> Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours. An explicit stack can store pending work instead. Both implement latest-pending-work first; a queue would change that discipline. Marking prevents infinite repetition around graph cycles.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 40-44

> III. Tree Traversing:
> Binary search tree can be traversed in three ways.
> a. Pre order traversal - traversing binary tree in the order of parent, left and right.
> b. Inorder traversal - traversing binary tree in the order of left, parent and right.
> c. Postorder traversal - traversing binary tree in the order of left, right and parent.

Question sources

Computer Science MCQ practice collection · Question 453

Question 188 | AM0455

Which specified sorting implementations are stable? Use strict adjacent swaps for bubble, ordinary in-place quicksort, left-first merge ties, and insertion shifts of strictly larger keys only. Select all that apply.

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A, C, D: A. Bubble sort; C. Merge sort; D. Insertion sort

Bubble sort's strict adjacent swaps do not exchange equal keys. Mergesort preserves ties by taking an equal key from the left run first. Insertion sort shifts only strictly larger keys, leaving equal earlier keys ahead of the inserted one. Those three mechanisms preserve relative equal-key order. Ordinary in-place quicksort does not guarantee it because partition exchanges can move equal records past one another.

Why each choice is right or wrong

A - Correct: Strict adjacent swaps preserve ties.

B - Incorrect: Ordinary quicksort does not guarantee equal-key order.

C - Correct: Left-first equal-key merging preserves ties.

D - Correct: Shifting only larger keys leaves equal earlier keys in place.

Primary note proof

Study-note section DSA32

> Strict swaps preserve equal-key order.

Study-note section DSA34

> Shift only strictly larger keys for stability.

Study-note section DSA37

> With left-first tie handling, equal keys retain order.

Study-note section DSA31

> stack usually log n average, n worst; unstable

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Computer Science MCQ practice collection · Question 455

Question 189 | AM0456

Which sorting algorithm is commonly used for small input sizes or partially sorted arrays?

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer D: Insertion sort

DSA34 explicitly favors insertion for small arrays or few inversions. Each inversion corresponds to shifting work, so nearly sorted data requires few shifts; sorted input is linear. This low-setup incremental algorithm is the conventional match among the choices. The other algorithms can sort these inputs, but their mere applicability does not identify the specific use case emphasized in the note.

Why each choice is right or wrong

A - Incorrect: Bubble can exploit early stopping but is not the note's specific small-array/few-inversions recommendation.

B - Incorrect: Quicksort incurs partition and recursive setup and is not the cited small-subarray method.

C - Incorrect: Ordinary merge sorting uses its merge levels and buffer even on nearly sorted input.

D - Correct: Insertion has low setup and few shifts on nearly sorted or small arrays.

Primary note proof

Study-note section DSA34

> Few inversions mean few shifts, favouring insertion for nearly sorted or small arrays. Shift only strictly larger keys for stability. A linked implementation saves array shifts but still must locate each insertion position.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 34-43

> A. Insertion Sort
> The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put

them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Computer Science MCQ practice collection · Question 456

Question 190 | AM0457

Which searching algorithm divides the search space into three parts at each step?

A. Linear search

B. Binary search

C. Ternary search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: Ternary search

DSA30 explicitly describes sorted-array ternary search as comparing two split points and retaining the relevant third. The three candidate regions give the technique its name. Binary search uses one midpoint and two regions, linear search scans successively, and DFS explores graph branches. Thus Ternary search uniquely matches the described three-part split.

Why each choice is right or wrong

A - Incorrect: Linear search advances through successive items and does not divide the remaining interval into thirds.

B - Incorrect: Binary search uses one midpoint and retains one of two halves, not one of three regions.

C - Correct: Ternary search uses two split points to divide the ordered search interval into three candidate parts.

D - Incorrect: DFS explores graph or tree branches rather than repeatedly trisecting an ordered search interval.

Primary note proof

Study-note section DSA30

> Sorted-array ternary search compares two split points and keeps the relevant third, still logarithmic. Jump search probes block boundaries then scans within a block. Regular intervals could mean unit-step scanning or larger jumps. Interpolation also needs sortedness, so prerequisite questions can overlap binary search.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

> Binary Searching
> • Works on ordered/sorted list
> • Uses divide and conquer method: it locates
> mid-point and identify which have the target

belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Question sources

Computer Science MCQ practice collection · Question 457

Question 191 | AM0458

Which ordinary array sorting implementations use O(1) total auxiliary space in both best and average cases? Count live recursive frames; use recursive quicksort without tail-call elimination and buffered mergesort. Select all that apply.

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A, D: A. Bubble sort; D. Insertion sort

The DSA31 table gives bubble and insertion constant extra storage across input cases. Their iterative forms retain only a fixed number of indices and temporaries, so total auxiliary space stays constant. The same table gives ordinary recursive quicksort a logarithmic average stack, and buffered mergesort needs a linear auxiliary array. Counting all live frames and buffers therefore leaves exactly bubble and insertion as the qualifying choices.

Why each choice is right or wrong

A - Correct: Bubble uses fixed temporary storage across cases.

B - Incorrect: The recursive quicksort stack is not constant in its ordinary average case.

C - Incorrect: Buffered mergesort uses linear auxiliary storage.

D - Correct: Insertion needs a saved key and fixed-size indices.

Primary note proof

Study-note section DSA32

For ascending order, compare each adjacent pair and swap only when left>right.

Study-note section DSA34

For each remaining item, save its value, shift larger preceding values right, and insert it into the gap.

Study-note section DSA31

stack usually log n average, n worst; unstable

Study-note section DSA37

Ordinary array merging needs a linear auxiliary buffer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put

them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Computer Science MCQ practice collection · Question 458

Question 192 | AM0459

Which searching algorithm is based on the principle of divide-and-conquer?

A. Linear search

B. Binary search

C. Hash search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B: Binary search

DSA29 compares the middle value with the target and continues only in the half that can contain it. This solves a smaller instance of the same ordered-search problem and is the divide-and-conquer reduction intended here. Linear scanning removes one candidate at a time, hashing computes addresses, and DFS follows adjacency without this ordered half-elimination rule.

Why each choice is right or wrong

A - Incorrect: Linear search scans successive candidates rather than repeatedly solving a half-size ordered-search instance.

B - Correct: Binary search divides the ordered interval and continues in the only half that can contain the target.

C - Incorrect: Hash search derives an address and resolves collisions rather than recursively dividing an ordered problem.

D - Incorrect: DFS follows adjacency branches and does not inherently apply the ordered half-elimination strategy described here.

Primary note proof

Study-note section DSA-S04

Divide and conquer reduces a problem to smaller instances of the same kind, solves the required smaller instances and obtains the original result from them. Mergesort recursively sorts two halves and merges; quicksort partitions and recursively sorts the resulting ranges. This course also classifies binary search as divide and conquer: an ordered comparison selects the only half that can contain the target. This is different from simply repeating adjacent swaps, choosing successive minima, or making stable LSD digit passes.

Study-note section DSA29

On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Computer Science MCQ practice collection · Question 459

Question 193 | AM0460

Which sorting algorithm has the worst-case time complexity of O(n log n)?

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: Merge sort

Ordinary merge combines n items over log n levels, giving n log n even in the worst case. The other offered algorithms have quadratic adverse-input work. Evidence: assigned PDF pages 4, 11, 3.

Why each choice is right or wrong

A - Incorrect: Bubble can perform quadratic passes/exchanges. See assigned PDF 4, 11, 11, 3.

B - Incorrect: Quicksort is explicitly quadratic in the worst case. See assigned PDF 4, 11, 11, 3.

C - Correct: Merge is explicitly n log n. See assigned PDF 4, 11, 11, 3.

D - Incorrect: Reverse-order insertion requires quadratic movement. See assigned PDF 4, 11, 11, 3.

Primary note proof

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Study-note section DSA36

> Balanced partitions give n log n work; repeated extreme pivots sum n+(n-1)+... to quadratic work.

Study-note section DSA34

> Reverse order causes 1+2+...+(n-1) shifts: quadratic time.

Study-note section DSA32

> Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Computer Science MCQ practice collection · Question 460

Question 194 | AM0461

Which sorting algorithm is efficient for sorting data in external memory (hard disks)?

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: Merge sort using buffered external runs.

DSA37 describes external sorting by building sorted runs and merging buffered disk streams to reduce I/O. Merge sort naturally supports that strategy even when the whole dataset does not fit RAM. Bubble and insertion sorting repeatedly move local data, while ordinary quicksort assumes partition access rather than the external sequential-run pattern specified here.

Why each choice is right or wrong

A - Incorrect: Bubble sort's repeated adjacent passes do not provide the buffered sorted-run strategy used for efficient external sorting.

B - Incorrect: Ordinary quicksort partitioning is not the sequential buffered-run merging strategy identified for external data here.

C - Correct: Merge sorting combines buffered sorted runs sequentially and is directly suited to disk-backed external sorting.

D - Incorrect: Insertion sort's repeated local insertion work does not provide the note's efficient external-run merging mechanism.

Primary note proof

Study-note section DSA37

> Top-down mergesort recurses; bottom-up merges runs of widths 1,2,4,... iteratively. In-place merging variants exist with different tradeoffs; linear merge names an operation, not a universally standardized sort variant. Internal sorting fits the working dataset in memory. External sorting builds sorted runs and merges buffered streams from disk, aiming to reduce I/O. Reverse order alone does not characterize maximum merge comparisons.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 38-44

> iv. Merge sort
> Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
> Algorithm:

1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Computer Science MCQ practice collection · Question 461

Question 195 | AM0462

A large dataset has an equality-key index implemented by a hash table with suitable key distribution and bounded load. No ordered or range queries are needed. Which listed search uses this index for expected constant-time lookup?

A. Linear search

B. Binary search

C. Hash search

D. Depth-first search

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: C. Hash search

Hash lookup computes a bucket or probe start from the key and uses the existing index to locate an equality match. With suitable distribution and bounded load, the note gives expected constant lookup. The important assumptions are the index and workload, rather than dataset size alone. Linear scanning ignores the index, binary search needs an ordered interval, and DFS follows graph relationships instead of this equality-key access structure.

Why each choice is right or wrong

A - Incorrect: A scan can inspect linearly many records and does not use the hash index.

B - Incorrect: Binary search uses an ordered interval rather than this index.

C - Correct: Hash search uses the stated index with expected constant lookup.

D - Incorrect: DFS follows graph connections rather than the equality-key index.

Primary note proof

Study-note section DSA26

Separate chaining stores colliding entries in a bucket list. Load alpha=n/m is average chain length; expected operations are O(1+alpha) under suitable distribution assumptions, with O(n) worst cases. Open addressing keeps entries inside the array and probes until a key or an available slot is found.

Study-note section DSA27

A dictionary or compiler symbol table can use hashing for expected constant lookup, or a balanced tree for logarithmic lookup with ordering. A bounded cache also needs eviction rules: FIFO uses a queue; LRU can combine hash lookup with a doubly linked recency list. Move a hit to the recent end and evict the least-recent end when full. No storage structure is universally best without operation and policy requirements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching
• Works on ordered/sorted list
• Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list
• The computational time for this algorithm is proportional to log2 n.

Question sources

Computer Science MCQ practice collection · Question 462

Question 196 | AM0463

For a uniform random permutation of distinct keys, which listed ordinary algorithms have the smallest tight average-case growth order? Use fixed-position-pivot quicksort, buffered mergesort without merge-skipping, and standard bubble/insertion. Select all algorithms tied for the smallest order.

A. Bubble sort

B. Quick sort

C. Merge sort

D. Insertion sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer B, C: B. Quick sort; C. Merge sort

Under the stated random-input model, ordinary quicksort has expected n log n work. Buffered mergesort performs n log n work in every order, so its average shares that order. Bubble and insertion have quadratic average work on random permutations. Quicksort and mergesort therefore tie for the smallest asymptotic average order among these options; the question does not ask for a measured running-time ranking between the tied methods. The DSA31 comparison table supplies the average-case entries used in this comparison.

Why each choice is right or wrong

A - Incorrect: Quadratic average growth is larger than n log n.

B - Correct: Expected n log n ties for the smallest listed average growth.

C - Correct: n log n across orders ties for the smallest listed average growth.

D - Incorrect: Quadratic average growth is larger than n log n.

Primary note proof

Study-note section DSA31

> n log n / n log n / n^2

Study-note section DSA31

> n / n^2 / n^2

Study-note section DSA37

> There are logarithmically many levels with linear merge work per level, giving n log n time even on sorted input unless an optimization skips merges.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 14-22

> ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
> Algorithm:
> 1. Choose a pivot value (mostly the first element is taken as the pivot value)
> 2. Position the pivot element and partition the list so that:
> • the left part has items less than or equal to the pivot value
> • the right part has items greater than or equal to the pivot value
> 3. Recursively sort the left part
> 4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Data Structure and Algorithms module | PDF page 3 | lines 34-43

A. Insertion Sort
The most instinctive type of sorting algorithm; it finds the location for an element and move all others up, and insert the element to its right position. It starts with the left most two items and put
them in order. Then add 3rd
item to the relatively sorted portion and find its correct location
(relative to each other) swapping adjacent elements if they are out of order. In such it sorts all item
in the list.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Computer Science MCQ practice collection · Question 463

Question 197 | AM0464

Which data structure is best suited for implementing a dictionary with fast lookup and insertion?

A. Array

B. Linked list

C. Hash table

D. Stack

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: Hash table for expected fast dictionary lookup and insertion.

DSA27 explicitly names hashing as a dictionary implementation with expected constant lookup. DSA26 qualifies this by a suitable distribution and controlled load; insertion is likewise expected fast under those conditions. Arrays and linked lists lack the same ordinary key-lookup performance, while a stack specifies LIFO access. This is an expected-performance recommendation, not a universal worst-case guarantee.

Why each choice is right or wrong

A - Incorrect: An ordinary array lacks key-derived lookup and may require scanning or maintaining a sorted order with costly insertions.

B - Incorrect: An ordinary linked list needs traversal to locate an arbitrary dictionary key, giving linear worst-case lookup.

C - Correct: A well-sized hash table offers expected fast key lookup and insertion under suitable hashing and load assumptions.

D - Incorrect: A stack exposes LIFO top operations and does not directly provide efficient arbitrary-key dictionary lookup.

Primary note proof

Study-note section DSA26

> Separate chaining stores colliding entries in a bucket list. Load alpha=n/m is average chain length; expected operations are O(1+alpha) under suitable distribution assumptions, with O(n) worst cases. Open addressing keeps entries inside the array and probes until a key or an available slot is found.

Study-note section DSA27

> A dictionary or compiler symbol table can use hashing for expected constant lookup, or a balanced tree for logarithmic lookup with ordering. A bounded cache also needs eviction rules: FIFO uses a queue; LRU can combine hash lookup with a doubly linked recency list. Move a hit to the recent end and evict the least-recent end when full. No storage structure is universally best without operation and policy requirements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.

Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 6 | lines 15-22

“A stack is an ordered list in which all insertions and deletions are made at one end, called the top”. Stacks are sometimes referred to as Last in First out (LIFO) lists. Stacks have some useful terminology associated with them:
• Push To add an element to the stack
• Pop To remove an element from the stock
• Peek To look at elements in the stack without removingthem
• LIFO Refers to the last in, first out behavior of thestack
• FILO (First in Last Out) : equivalent to LIFO

Question sources

Computer Science MCQ practice collection · Question 464

Question 198 | AM0465

Which listed algorithm guarantees shortest paths from one source to all reachable vertices for arbitrary graphs with nonnegative edge weights that may differ? Compare the standard forms of the listed algorithms.

A. Depth-first search

B. Breadth-first search

C. Dijkstra's algorithm

D. Prim's algorithm

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer C: C. Dijkstra's algorithm

Standard Dijkstra repeatedly finalizes the unsettled vertex with the smallest tentative distance and relaxes its outgoing edges. Nonnegative edge weights justify this shortest-path procedure even when different edges have different costs. BFS minimizes edge count and is sufficient when edge costs are equal, which is weaker than the stated guarantee. DFS supplies reachability paths, while Prim minimizes total spanning-tree cost rather than source-to-vertex route lengths.

Why each choice is right or wrong

A - Incorrect: A DFS path need not have minimum total weight.

B - Incorrect: BFS minimizes edge count and does not guarantee minimum weight for arbitrary differing costs.

C - Correct: Dijkstra guarantees source shortest paths under nonnegative weights.

D - Incorrect: Prim solves the spanning-tree objective rather than source shortest paths.

Primary note proof

Study-note section DSA43

> For nonnegative weights, standard Dijkstra initializes source distance zero and others infinity, repeatedly finalizes the unsettled vertex with smallest distance, and relaxes each outgoing edge: dist[v]=min(dist[v],dist[u]+w(u,v)). A min-priority queue supports the next choice. BFS is enough when every edge has equal cost; negative edges require another justified method, such as Bellman-Ford.

Study-note section DSA41

> BFS finds minimum-edge paths in unweighted graphs, not arbitrary minimum-weight paths.

Question sources

Computer Science MCQ practice collection · Question 465

Question 199 | AM0466

Which listed standard representations directly implement stack push/pop at an array top or linked-chain head, without simulating LIFO through another ADT? Select all that apply.

A. Array

B. Linked list

C. Queue

D. Tree

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A, B: A. Array; B. Linked list

The array representation maintains a top index and changes it when pushing or popping. The linked representation inserts and removes at the chain head, so the latest node is also the next one removed. Both directly implement LIFO with ordinary constant endpoint operations. A queue has FIFO service, while a generic tree has no stack-top contract; emulations through those ADTs are outside this representation question.

Why each choice is right or wrong

A - Correct: An array with a top index directly implements stack operations.

B - Correct: A linked chain with head insertion/removal directly implements stack operations.

C - Incorrect: An ordinary FIFO queue is not a direct stack representation.

D - Incorrect: A generic tree does not supply the specified stack-top representation.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA01

> Arrays or linked nodes can implement the same LIFO stack.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 24-42

> I. Array implementation (static memory).
> Push(n): Adding item at top
> top is the current top of stack and n is its
> maximum size

```
Begin
if (top == n) then stack is full;
else
top := top+1;
stack[top] := item;
end:
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;
```

Data Structure and Algorithms module | PDF page 6 | lines 43-45

II. Linked list (dynamic memory)
The linked list implementation of stack is adding items at the end of the chain (Push item) and deleting the end node of the list (Pop item).

Question sources

Computer Science MCQ practice collection · Question 466

Question 200 | AM0467

Given a consistent comparator on a priority field, which listed comparison sorting algorithms can sort records by that field? No particular extraction mechanism or equal-priority stability is required. Select all that apply.

A. Bubble sort

B. Quick sort

C. Merge sort

D. Heap sort

Hint

Compare the algorithmic steps and access policy with the exact resource or ordering requirement in the question.

Answer A, B, C, D: A. Bubble sort; B. Quick sort; C. Merge sort; D. Heap sort

Priority is a key field that any comparison sort can use through a consistent comparator. Bubble compares neighboring priorities, quicksort partitions by priority, mergesort combines priority-ordered runs, and heapsort repeatedly places a priority extreme. Since no heap-extraction mechanism or tie-stability condition is required, every listed algorithm can produce the requested priority ordering. Priorities alone do not identify one unique sorting method.

Why each choice is right or wrong

A - Correct: Bubble can compare adjacent priority fields.

B - Correct: Quicksort can partition with the priority comparator.

C - Correct: Mergesort can merge runs by the priority field.

D - Correct: Heapsort can extract priority extremes.

Primary note proof

Study-note section DSA38

> It is not uniquely the method for sorting by priorities: any comparison sort can compare a priority field. Heap extraction is its distinctive mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 23-37

> iii. Heap Sort
> Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
> each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
> Algorithm:
> 1. Construct a binary tree
> • The root node corresponds to Data[0].
> • If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
> 2. Construct the heap tree from initial binary tree using "adjust" process.
> Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Data Structure and Algorithms module | PDF page 8 | lines 40-45

> Priority Queue:- is a queue where each data has an associated key provided at insertion time.
> Application of Queue
> Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:

• Print server- maintains a queue of print jobs
• Disk Driver- maintains a queue of disk input/output requests
• Task scheduler in multiprocessing system- maintains priority queues of processes

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Data Structure and Algorithms module | PDF page 4 | lines 2-8

C. Bubble Sort
The simplest to implement but slowest algorithm on very large inputs. The algorithm loops through
the array from i=0 to n and swap adjacent elements if they are out of order bubbling or pushing out the smallest or largest element to its proper position.
• How many comparisons? => O(n2
) and How many swaps? => O(n2
)

Question sources

Computer Science MCQ practice collection · Question 467

Question 201 | AM0468

An LRU cache already has hash lookup that returns a reference to the matching entry. Which listed representation should maintain recency order so a known entry can be removed and moved to the recent end in constant time, with eviction at the least-recent end?

A. Array

B. Doubly linked list with head and tail

C. Hash table

D. Queue

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer B: B. Doubly linked list with head and tail

The hash index handles locating an entry, while the recency component must unlink a known entry and reconnect it at the recent end. A doubly linked list supplies both neighboring links for that constant-time removal and maintains endpoints for movement and eviction. An array would generally shift entries, a bare hash table does not supply recency order, and an ordinary FIFO queue cannot remove an arbitrary known interior entry.

Why each choice is right or wrong

A - Incorrect: Maintaining array recency order generally requires shifts.

B - Correct: A doubly linked recency list supports unlinking a known node and endpoint updates.

C - Incorrect: The existing hash index handles lookup but does not itself maintain LRU order.

D - Incorrect: An ordinary queue lacks arbitrary known-interior-node removal.

Primary note proof

Study-note section DSA27

> A dictionary or compiler symbol table can use hashing for expected constant lookup, or a balanced tree for logarithmic lookup with ordering. A bounded cache also needs eviction rules: FIFO uses a queue; LRU can combine hash lookup with a doubly linked recency list. Move a hit to the recent end and evict the least-recent end when full. No storage structure is universally best without operation and policy requirements.

Study-note section DSA10

> Doubly linked nodes add prev; head and tail enable constant-time operations at both ends. Singly linked tail deletion still needs its predecessor. A circular tail links back to head; stop traversal on returning to the start, handling empty separately.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 4 | lines 16-25

1. Single Linked lists
It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
Basic Operations on Single Linked List
I. Creating Linked Lists
Structure definition of a single linked list need to contain both data items declaration and next node
pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node added
to the chain.

Data Structure and Algorithms module | PDF page 5 | lines 13-24

Step 3: Then add the node at the end of the list. For this we need to check whether the list is empty or not. If list is empty the new node will be the first (Start) node, otherwise navigate forward starting form the first node till last node found and add it next to this node which make the new node the last node.
if (start == NULL) // If list is empty, then new node become start node
start = temp;
else {
temp2 = start; // list not empty! Thus temp2 starts from first node
while (temp2->next!= NULL) { // till the current last node found navigates forward
temp2 = temp2->next; // Move to next link in chain
}
temp2->next = temp; // new node (temp) appended at the end of the list

Data Structure and Algorithms module | PDF page 5 | lines 35-41

IV. Deleting a node from the list
When a node is deleted, the space that it took up should be reclaimed. Otherwise the computer will eventually run out of memory space. This is done with the delete instruction:
delete temp; // Release the memory pointed to by temp
However, we can't just delete the nodes willy-nilly as it would break the chain. We need
to reassign the pointers and then delete the node at the last moment. We can delete nodes
at the start, middle or end of the list. Here let’s see the steps to delete at the end.

Data Structure and Algorithms module | PDF page 7 | lines 10-16

“A queue is an ordered list in which all insertions at one end called REAR and deletions are made at another end called FRONT”. Queues are sometimes referred to as First in First out (FIFO) lists.
The two basic operations are:
Enqueue - Inserts an item / element at the rear end of the queue. An overflow error occurs
if the queue is full.
Dequeue - Removes an item / element from the front end of the queue, and returns it to
the user. An underflow error occurs if the queue is empty.

Question sources

Computer Science MCQ practice collection · Question 468

Question 202 | AM0469

Which algorithm is used to find the maximum subarray sum in an array?

A. Binary search

B. Quick sort

C. Merge sort

D. Kadane's algorithm

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer D: Kadane’s algorithm

DSA44 directly identifies Kadane as the contiguous maximum-sum-subarray algorithm. It updates the best sum ending at the current item and the best sum seen overall, using linear time and constant working storage. Binary search does not optimize contiguous sums, and quicksort or merge sort reorder values rather than preserve the original adjacency needed by this problem.

Why each choice is right or wrong

A - Incorrect: Binary search locates targets or ordered boundaries; it does not compute a contiguous maximum-sum subarray.

B - Incorrect: Quicksort reorders values and does not preserve the original adjacency required for maximum subarray sum.

C - Incorrect: Merge sort orders values; the named sorting algorithm is not the maximum-subarray recurrence.

D - Correct: Kadane maintains the best sum ending here and the best anywhere, directly solving the stated problem.

Primary note proof

Study-note section DSA44

> Kadane’s algorithm finds a contiguous maximum-sum subarray. For a required nonempty answer, initialize ending=best=A[0]; update ending=max(x,ending+x), then best=max(best,ending). It is linear time with constant working storage. Initializing best=0 instead silently allows an empty answer and fails the nonempty all-negative case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching
• Works with unsorted List
• Loop through the list starting at the first element
• Time is proportional to the size of input (n) and we call this time complexity O(n).

Data Structure and Algorithms module | PDF page 11 | lines 14-22

ii. Quick sort: is the fastest known algorithm. It uses divide and conquer strategy and in the worst case its complexity is O (n2). But its expected complexity is O(nlogn).
Algorithm:
1. Choose a pivot value (mostly the first element is taken as the pivot value)
2. Position the pivot element and partition the list so that:
• the left part has items less than or equal to the pivot value
• the right part has items greater than or equal to the pivot value
3. Recursively sort the left part
4. Recursively sort the right part

Data Structure and Algorithms module | PDF page 11 | lines 38-44

iv. Merge sort
Merge sort uses divide and conquer strategy and its time complexity is O(nlogn).
Algorithm:
1. Divide the array in to two halves.
2. Recursively sort the first n/2 items.
3. Recursively sort the last n/2 items.
4. Merge sorted items (using an auxiliary array).

Question sources

Computer Science MCQ practice collection · Question 469

Question 203 | AM0470

Which data structure is commonly used to implement a graph?

A. Array

B. Linked list

C. Hash table

D. Adjacency list

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer D: Adjacency list

DSA40 describes adjacency lists as storing each vertex's neighbors in Θ(V+E) space. This names a complete graph representation among the choices. Arrays, linked lists, or maps may store parts of a representation, but those container names alone do not specify how vertices and their edges are encoded. The answer identifies the graph-specific organization.

Why each choice is right or wrong

A - Incorrect: An array can store a matrix or adjacency containers, but its name alone does not specify a graph representation.

B - Incorrect: A linked list can hold neighbor records, but alone does not describe the vertex-to-neighbor organization.

C - Incorrect: Hash tables may index vertices or neighbors, but do not by themselves name the graph representation supplied here.

D - Correct: An adjacency list explicitly associates vertices with their neighbors and is a standard graph representation.

Primary note proof

Study-note section DSA40

> A matrix uses Theta(V^2) space and constant edge lookup. Adjacency lists use Theta(V+E) space and enumerate neighbours. Dense simple graphs approach quadratic edge counts. Examining every edge costs more on denser graphs; vertex-only operations need not inspect every edge.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 2 | lines 15-18

> a. Linear data structure: data items stores in sequence
> Example: Array, stack and Queue,
> b. Non-Linear data structures : in which there is order of data items
> Example: Trees and Graphs

Data Structure and Algorithms module | PDF page 4 | lines 16-25

> 1. Single Linked lists
> It is a self-referential structure (Structures can hold pointers to instances of themselves), that allow
> to create a chain of structs/Nodes with related data. Each node in the list contains data items and pointer to next node.
> Basic Operations on Single Linked List
> I. Creating Linked Lists
> Structure definition of a single linked list need to contain both data items declaration and next node
> pointer. In addition the first node pointer is essential to keep track of the list to perform various operations on the list. Thus, it is initialized to NULL first as the list is empty before any node

added
to the chain.

Question sources

Computer Science MCQ practice collection · Question 470

Question 204 | AM0471

Which algorithm is used to find the minimum spanning tree in a graph?

A. Depth-first search

B. Breadth-first search

C. Dijkstra's algorithm

D. Prim's algorithm

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer D: Prim’s algorithm

DSA43 defines an MST as a minimum-total-cost connection of all vertices in a connected undirected weighted graph. Prim repeatedly adds the cheapest edge crossing from its current tree to an outside vertex. Dijkstra minimizes source-path distances instead, while BFS and DFS are traversals without a general minimum-weight spanning-tree guarantee. Thus Prim is the listed MST algorithm.

Why each choice is right or wrong

A - Incorrect: DFS can produce a spanning tree but does not generally minimize total edge weight.

B - Incorrect: BFS produces minimum-edge source layers in unweighted graphs, not a general minimum-weight spanning tree.

C - Incorrect: Dijkstra minimizes source-to-vertex distances, a different objective from total spanning-tree connection cost.

D - Correct: Prim grows a spanning tree by repeatedly choosing a cheapest edge crossing to an outside vertex.

Primary note proof

Study-note section DSA43

An MST minimizes the total cost of connecting all vertices of a connected undirected weighted graph. Prim adds the cheapest edge crossing from its current tree to an outside vertex. Kruskal processes edges by increasing weight and rejects those joining vertices already connected. An MST need not give shortest source-to-vertex routes, and equal weights can permit several MSTs.

Study-note section DSA43

For nonnegative weights, standard Dijkstra initializes source distance zero and others infinity, repeatedly finalizes the unsettled vertex with smallest distance, and relaxes each outgoing edge: dist[v]=min(dist[v],dist[u]+w(u,v)). A min-priority queue supports the next choice. BFS is enough when every edge has equal cost; negative edges require another justified method, such as Bellman-Ford.

Question sources

Computer Science MCQ practice collection · Question 471

Question 205 | AM0472

A priority queue needs O(1) peek and O(log n) worst-case insertion and extraction for an interleaved workload. Compare ordinary sorted/unsorted array and linked-list implementations, a binary heap, and an unmodified FIFO queue; exclude representations with extra search indexes. Which meets all requirements?

A. Array

B. Linked list

C. Binary heap

D. Queue

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer C: C. Binary heap

A binary heap exposes the extreme key at its root, so peek is constant, and insertion or extraction restores order along a logarithmic-height path. Ordinary unordered arrays or lists need a linear search for an extreme, while ordered versions pay linear insertion in the general case. An unmodified FIFO queue follows arrival order rather than priority. The stated operation bounds make the heap the unique matching standard representation.

Why each choice is right or wrong

A - Incorrect: Ordinary ordered/unsorted arrays trade linear insertion against linear extremum selection.

B - Incorrect: Ordinary lists likewise have a linear cost in one required operation.

C - Correct: A binary heap provides constant peek and logarithmic insertion/extraction.

D - Incorrect: An unmodified queue serves FIFO rather than priority order.

Primary note proof

Study-note section DSA17

A priority queue returns the smallest or largest priority key under its convention. Urgent patients may precede earlier arrivals. Insertions and extractions can interleave. Binary heaps support constant peek and logarithmic insert/extract.

Study-note section DSA17

Arrays and lists can also implement priority queues: an unordered sequence gives cheap insertion but linear extremum removal; an ordered sequence reverses that tradeoff. Equal priorities have no inherent FIFO guarantee. If stable service is required, compare (priority, arrival_counter). Choose the priority direction and tie rule before tracing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 11 | lines 23-37

iii. Heap Sort
Heap sort operates by first converting the list in to a heap tree. Heap tree is a binary tree in which
each node has a value greater than both its children (if any). It uses a process called "adjust to accomplish its task (building a heap tree) whenever a value is larger than its parent. The time complexity of heap sort is O(nlogn).
Algorithm:
1. Construct a binary tree
• The root node corresponds to Data[0].
• If we consider the index associated with a particular node to be i, then the left child of this node corresponds to the element with index 2*i+1 and the right child corresponds to the element with index 2*i+2. If any or both of these elements do not exist in the array, then the corresponding child node does not exist either.
2. Construct the heap tree from initial binary tree using "adjust" process.
Sort by swapping the root value with the lowest, right most value and deleting the lowest, right most value and inserting the deleted value in the array in it proper position.

Data Structure and Algorithms module | PDF page 8 | lines 40-45

Priority Queue:- is a queue where each data has an associated key provided at insertion time.
Application of Queue
Queue is goodfor fair (firstcome first served) ordering of actions. Some of queue applications:
• Print server- maintains a queue of print jobs
• Disk Driver- maintains a queue of disk input/output requests
• Task scheduler in multiprocessing system- maintains priority queues of processes

Question sources

Computer Science MCQ practice collection · Question 472

Question 206 | AM0473

Which algorithm is commonly used for pattern matching in strings?

A. Depth-first search

B. Breadth-first search

C. Dijkstra's algorithm

D. Knuth-Morris-Pratt algorithm

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer D: Knuth-Morris-Pratt

DSA44 explicitly defines KMP as finding a pattern inside text. It preprocesses longest proper prefix/suffix lengths and uses them to avoid restarting comparisons after mismatch, giving linear combined pattern-and-text work. DFS, BFS, and Dijkstra concern graph exploration or paths, rather than this specialized substring-matching mechanism. KMP uniquely matches the question among these choices.

Why each choice is right or wrong

A - Incorrect: DFS explores graph or tree branches; it is not the substring-pattern algorithm identified in DSA44.

B - Incorrect: BFS explores graph distance layers rather than applying prefix/suffix reuse for text matching.

C - Incorrect: Dijkstra computes nonnegative weighted shortest paths, not substring pattern occurrences.

D - Correct: KMP is the pattern-matching algorithm using preprocessed prefix/suffix lengths to avoid repeated text comparisons.

Primary note proof

Study-note section DSA44

> Knuth-Morris-Pratt (KMP) finds a pattern inside text. Preprocess the pattern’s longest proper prefix that is also a suffix. After mismatch, fall back using those prefix lengths rather than restarting comparisons from the beginning; the text pointer need not move backward. Pattern preprocessing plus scanning takes O(pattern length+text length).

Question sources

Computer Science MCQ practice collection · Question 473

Question 207 | CS0122

Assume a very large finite key universe U but only n stored keys, with n much smaller than |U|. A hash table uses capacity proportional to n. Compare allocated storage, rather than unconditional speed. Which statement is false?

A. For n much smaller than |U|, the specified hash table uses less allocated storage than direct addressing.

B. Direct Address table doesn't use hash function to map keys

C. Hash table allocates one separate memory slot for each key in the universe U

D. There is one-to-one correspondence between keys in universe U and memory slots in the Direct Address table

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer C: C. Hash table allocates one separate memory slot for each key in the universe U

Direct addressing reserves a distinct slot for every possible key in the universe, whether present or absent, so its storage is proportional to |U|. The specified hash table stores the actual entries plus capacity proportional to n. For this sparse-universe storage comparison, hashing uses less space. C incorrectly attributes the direct-address allocation rule to hashing, while direct key-to-slot mapping and separate universe slots accurately describe direct addressing.

Why each choice is right or wrong

A - Incorrect: Under the sparse-universe storage assumptions, the hash table uses fewer allocated slots.

B - Incorrect: Direct addressing maps directly to a key’s universe slot without compressive hashing.

C - Correct: One slot for every possible key describes direct addressing, not the specified hash table.

D - Incorrect: Each universe key has a separate direct-address slot.

Primary note proof

Study-note section DSA25

> Direct addressing reserves a position for every possible key in a finite universe U, costing Theta(|U|) slots. A hash table uses h(key) in 0..m-1 and stores only actual entries plus bucket capacity. With h(k)=k mod 6, key 36 has home address 0; collisions may move its final storage position.

Question sources

Computer Science Exit Exam practice collection · Question 122

Question 208 | CS0131

What is the time complexity order of push() operation of Stack data structure implemented using array?

A. O(n)

B. O(log n)

C. O(n^2)

D. O(1)

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer D: O(1)

The fixed-array push checks fullness, increments top and stores one item. A bounded number of unit-cost operations independent of n gives O(1). Evidence: assigned PDF pages 6.

Why each choice is right or wrong

A - Incorrect: Linear would require scanning or copying that the given push does not do. See assigned PDF 6.

B - Incorrect: Logarithmic would require a search/adjustment absent from push. See assigned PDF 6.

C - Incorrect: Quadratic has no nested work in this code. See assigned PDF 6.

D - Correct: Only a fullness test, index increment and one assignment are performed. See assigned PDF 6.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA11

> A count-based alternative uses size=0 and A[size++]=x. Do not combine its empty/full tests with the top-index convention. Fixed-array push/pop are constant time. Linked stacks achieve constant time by inserting and deleting at the head; using a singly linked tail for pop would require a predecessor traversal.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 6 | lines 24-42

I. Array implementation (static memory).
Push(n): Adding item at top
top is the current top of stack and n is its
maximum size
Begin
if (top == n) then stack is full;
else
top := top+1;
stack[top] := item;
end:
Pop(): removing item from stack top
remove top element from the stack and put it in
the item
Begin:
if (top == 0) then stack is empty;
else
item := stack[top];
top := top-1;
end;

Question sources

Computer Science Exit Exam practice collection · Question 131

Question 209 | CS0132

A zero-based six-slot hash table uses h(key)=key%6. What home index does it compute for key 36, before any collision resolution?

A. At index 35

B. At index 1

C. At index 6

D. At index 0

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer D: D. At index 0

The home address is obtained by applying the stated remainder function. Since 36 is exactly divisible by 6, the remainder is zero and the home index is 0. The question stops before collision resolution, so occupied slots cannot change this computed address. Index 1 could be a displaced location under some probing state, but it is not the function's output; indices 6 and 35 are outside the table.

Why each choice is right or wrong

A - Incorrect: 35 is outside the valid zero-based range 0-5.

B - Incorrect: 1 is not the remainder of 36 divided by 6.

C - Incorrect: 6 is outside the valid range and is not a possible remainder modulo 6.

D - Correct: 36%6 equals 0.

Primary note proof

Study-note section DSA25

> Direct addressing reserves a position for every possible key in a finite universe U, costing Theta(|U|) slots. A hash table uses h(key) in 0..m-1 and stores only actual entries plus bucket capacity. With h(k)=k mod 6, key 36 has home address 0; collisions may move its final storage position.

Question sources

Computer Science Exit Exam practice collection · Question 132

Question 210 | CS0133

Which method can resolve collision in hashed files by proceeding from the occupied position specified by the hash address and checking subsequent positions until an empty position is found?

A. Open addressing

B. Multiple hashing

C. Chaining

D. Cluster indexing

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer A: Open addressing, specifically linear probing.

DSA26 defines open addressing as keeping entries within the array and probing until an available slot is found. Checking successive positions is its linear-probing form. Multiple hashing changes probe steps, chaining stores collisions in bucket lists, and cluster indexing is not this insertion procedure. Therefore Open addressing is the matching general method among the options.

Why each choice is right or wrong

A - Correct: Open addressing searches alternative table positions; checking successive positions is its linear-probing form.

B - Incorrect: Multiple or double hashing derives another probe step rather than the stated simple successive-slot scan.

C - Incorrect: Chaining puts colliding entries in a bucket list instead of searching successive array slots.

D - Incorrect: Cluster indexing does not name the collision insertion strategy described by the successive-position probing rule.

Primary note proof

Study-note section DSA26

> Separate chaining stores colliding entries in a bucket list. Load alpha=n/m is average chain length; expected operations are O(1+alpha) under suitable distribution assumptions, with O(n) worst cases. Open addressing keeps entries inside the array and probes until a key or an available slot is found.

Study-note section DSA26

> Linear probing tries (h(k)+j)%m. For size 7 and keys 10,17,24, all home addresses are 3 and occupied positions become 3,4,5. Quadratic probing changes the step; double hashing uses h1(k)+j h2(k), modulo m, with a step relatively prime to m to cover all slots. Neither method guarantees no collisions. Deletion needs tombstones or reinsertion so searches do not stop prematurely.

Question sources

Computer Science Exit Exam practice collection · Question 133

Question 211 | CS0134

Which one of the following is true about tree data structure?

A. Tree is a static data structure

B. Tree is a linear data structure

C. Tree forms a cycle - in other words tree is a cyclic data structure

D. There is only one unique path when we traverse from one node to another node in a tree

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer D: There is only one unique path when we traverse from one node to another node in a tree

A rooted tree gives every nonroot node exactly one parent and a unique root path. Between any two nodes, ascend to their common ancestor and descend; a second distinct path would create a cycle and violate the parent/path hierarchy. Evidence: assigned PDF pages 9, 2.

Why each choice is right or wrong

A - Incorrect: The pointer-based tree representation does not require static size. See assigned PDF 9, 2, 9.

B - Incorrect: The source explicitly classifies trees as nonlinear. See assigned PDF 9, 2, 9.

C - Incorrect: A cycle contradicts the unique-path hierarchy. See assigned PDF 9, 2, 9.

D - Correct: The rooted hierarchy gives one unique connecting path for every node pair. See assigned PDF 9, 2, 9.

Primary note proof

Study-note section DSA18

> A rooted tree has one parentless root and one parent per other node. It is connected and acyclic, has n-1 edges, and one path between any pair. Depth counts root-to-node edges; height is maximum depth. A single-node tree therefore has height zero. Leaves have no children.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 9 | lines 3-13

> A tree is a set of nodes and edges that connect pairs of nodes that connect pairs of nodes. It is an abstract model of a hierarchical structure. Rooted tree has the following structure:
> • One node distinguished as root.

• Every node C except the root is connected from exactly other node P. P is C's parent, and C is one of C's children.
• There is a unique path from the root to the each node.
• The number of edges in a path is the length of the path.
Tree Terminologies:
• Root: a node without a parent.
• Internal node: a node with at least one child.
• External (leaf) node: a node without a child.

Data Structure and Algorithms module | PDF page 2 | lines 15-18

a. Linear data structure: data items stores in sequence
Example: Array, stack and Queue,
b. Non-Linear data structures : in which there is order of data items
Example: Trees and Graphs

Data Structure and Algorithms module | PDF page 9 | lines 19-25

• Binary tree: a tree in which each node has at most two children called left child and right child.
• Binary search tree (ordered binary tree): a binary tree that may be empty, but if it is not empty it
satisfies the following.
o Every node has a key and no two elements have the same key.
o The keys in the right subtree are larger than the keys in the root.
o The keys in the left subtree are smaller than the keys in the root.
o The left and the right subtrees are also binary search trees.

Question sources

Computer Science Exit Exam practice collection · Question 134

Question 212 | CS0135

For an unordered heap record file without an ordered index or a maintained sorted copy, which statement is not correct? Ordinary insertion uses available storage rather than maintaining field order.

A. Sorting is required to read records in particular field order

B. Linear search can be used to search a necessary record

C. Record insertion is efficient

D. New records are inserted at the beginning of the file

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer D: D. New records are inserted at the beginning of the file

A heap file does not maintain record order by a chosen field. With no ordered index or sorted copy, ordered output therefore requires sorting; an ordinary scan can locate a record, and insertion can place a record in available storage. Nothing requires new records to be inserted at the beginning of the file. D is the incorrect claim, while the explicit no-index scope makes A's ordering requirement appropriate.

Why each choice is right or wrong

A - Incorrect: Under the specified no-index/no-sorted-copy scope, field-ordered output requires ordering the records.

B - Incorrect: A linear scan can search unordered records.

C - Incorrect: Using available storage avoids maintaining a sorted record sequence during insertion.

D - Correct: Heap-file insertion is not required to place every new record at the beginning.

Primary note proof

Study-note section DSA45

> Specify whether a question asks which structure can implement something or which is best under stated costs. ADTs do not force one representation, and serial-access memory is not inherently LIFO. A heap file is an unordered database record file, unrelated to a binary heap: insertion uses available space, scanning can find records, and ordered output requires sorting or an index. It need not insert at the beginning.

Question sources

Computer Science Exit Exam practice collection · Question 135

Question 213 | NS0298

Which listed structures are linear in the course classification? Select all that apply.

A. Tree

B. Array

C. Queue

D. Linked list

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer B, C, D: B. Array; C. Queue; D. Linked list

A linear structure arranges items along a sequence rather than through hierarchical branching. The note explicitly includes arrays, queues and linked lists in that category, giving three valid selections. A tree instead models a hierarchy and is classified as nonlinear. The select-all instruction allows every listed member of the linear category to be recognized rather than forcing an arbitrary single choice.

Why each choice is right or wrong

A - Incorrect: Trees model hierarchy rather than a linear sequence.

B - Correct: Arrays are classified as linear.

C - Correct: Queues are classified as linear.

D - Correct: Linked lists are classified as linear.

Primary note proof

Study-note section DSA01

> An abstract data type (ADT) defines values and operations; representation supplies implementation. Arrays or linked nodes can implement the same LIFO stack. Abstraction keeps relevant details: patient urgency and arrival order matter for service; hair colour does not.

Question sources

Computer Science Exit Exam · 2018 · Question 15

Question 214 | NS0319

Which abbreviations name the stack service policy? Select all that apply.

A. LIFO

B. FIFO

C. FILO

D. FCFS

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer A, C: A. LIFO; C. FILO

The most recently pushed item is removed first by a stack. This is last in, first out, abbreviated LIFO. Equivalently, the first item inserted remains until later items have been removed, making it first in, last out or FILO. FIFO and FCFS instead preserve arrival order in ordinary queue service. Both A and C therefore name the same stack discipline.

Why each choice is right or wrong

A - Correct: LIFO means last in, first out.

B - Incorrect: FIFO means first in, first out, an ordinary queue policy.

C - Correct: FILO is the equivalent first in, last out view of a stack.

D - Incorrect: FCFS describes arrival-order service rather than stack service.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA14

> An ordinary queue enqueues at rear, dequeues at front, and peeks at front. This is FIFO or first-come, first-served. A linked implementation with head and tail supports both operations in constant time: append at tail, remove at head. The first insertion sets both pointers; removing the last item clears both. With head alone, insertion must traverse to the tail.

Question sources

Computer Science Exit Exam · 2018 · Question 44

Question 215 | NS0337

For ordinary binary search on a sorted random-access array of n elements, what is the tightest listed worst-case time bound? Count each indexed probe as constant work.

A. $O(n (\log n)^2)$

B. $O(n^2)$

C. O(n)

D. O(log n)

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer D: D. O(log n)

Binary search repeatedly removes roughly half of the remaining candidate interval. The note bounds its worst probes by $\text{floor}(\log_2 n)+1$ for nonempty input, giving logarithmic worst-case time under constant-cost indexing. The linear, quadratic and n-times-squared-log options all grow faster and are looser upper bounds. Explicitly asking for the tightest bound selects the logarithmic alternative rather than treating every valid Big-O upper bound as equally responsive.

Why each choice is right or wrong

A - Incorrect: $n(\log n)^2$ is a much looser upper bound.

B - Incorrect: Quadratic is a looser upper bound.

C - Incorrect: Linear is a looser upper bound.

D - Correct: Logarithmic matches the interval-halving probe growth.

Primary note proof

Study-note section DSA29

> Worst probes are $\text{floor}(\log_2 n)+1$ for $n>0$: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Study-note section DSA06

> For eventually nonnegative functions, $f(n)=O(g(n))$ means $f(n) \le c\, g(n)$ for some positive c beyond a threshold k. Omega reverses the inequality; Theta requires both. Little-o means f/g tends to zero; little-omega means it grows without bound. These relations can describe best-case, average-case, worst-case or space functions.

Question sources

Computer Science Exit Exam · 2018 · Question 68

Question 216 | NS0351

Which sorting algorithm compares each pair of adjacent elements from the beginning of an array and, if they are in reversed order, swaps them; if at least one swaps has been done, repeat step 1.

A. Insertion sort

B. Bubble sort

C. Selection sort

D. Shell sort

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer B: Bubble sort with early stopping

DSA32 describes adjacent comparisons, swapping strict inversions, and repeating passes until a pass performs no swaps. That matches every part of the stem, including its stopping condition. Insertion grows a sorted prefix, selection chooses a suffix minimum, and Shell uses separated gap subsequences, so none has this exact adjacent full-pass mechanism.

Why each choice is right or wrong

A - Incorrect: Insertion places each next value into a sorted prefix rather than repeating full adjacent passes.

B - Correct: Adjacent inversion swaps and repetition until no swaps identify optimized bubble sort.

C - Incorrect: Selection finds a suffix minimum and exchanges it with the prefix boundary.

D - Incorrect: Shell sorts gap-separated subsequences before its final gap-one insertion pass.

Primary note proof

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right. The largest remaining value moves to the final unsorted position after one left-to-right pass. Shrink the next pass by one position. Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Question sources

Computer Science Exit Exam · 2018 · Question 87

Question 217 | NS0354

Among the following one is NOT the application of queue in the real world.

A. It is used in operating systems for handling interrupts.

B. It is used to maintain the play list in media players in order

C. It is used in Depth First search

D. It is widely used as waiting lists for a single shared resource

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer C: Depth-first search uses stack discipline rather than the ordinary queue.

DSA42 finishes a branch using a stack or recursion and explicitly says a queue changes that discipline. DSA14 gives FIFO waiting, play sequences, and asynchronous buffering as queue uses. Interrupt work can also be buffered pending service. Thus the standard DFS claim is the non-queue application among the offered ordinary examples.

Why each choice is right or wrong

A - Incorrect: Pending interrupts can be buffered for later service; queue use need not imply one universal interrupt priority policy.

B - Incorrect: A FIFO play queue can preserve the order of media items.

C - Correct: Standard DFS uses latest-pending-work-first stack or recursion rather than FIFO queue service.

D - Incorrect: Waiting for one shared resource is a direct FIFO queue application.

Primary note proof

Study-note section DSA42

> Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours. An explicit stack can store pending work instead. Both implement latest-pending-work first; a queue would change that discipline. Marking prevents infinite repetition around graph cycles.

Study-note section DSA14

> FIFO suits equal-priority printer jobs, shared-resource waiting, asynchronous buffers and level-order exploration. A play sequence can also use a queue. Distinguish airline seat records from waiting requests; an application name alone does not force one representation.

Study-note section DSA-S02

> An interrupt service routine may need to record work quickly and defer its completion. Operating systems can queue that deferred work for later processing; Windows deferred procedure calls are one concrete example. This is a queue application, but it does not imply that every interrupt system uses a single FIFO service policy.

Question sources

Computer Science Exit Exam · 2018 · Question 91

Question 218 | NS0519

Which data structure uses LIFO?

A. Queue

B. Array

C. Stack

D. Linked List

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer C: Stack

DSA11 explicitly assigns LIFO to a stack. Arrays and linked lists can represent that stack, but neither storage structure inherently fixes its removal order. A queue instead supplies FIFO. The question asks for the service-policy ADT, so Stack is the specific answer rather than one of the possible underlying representations.

Why each choice is right or wrong

A - Incorrect: A queue follows FIFO, not the latest-in-first-out order requested.

B - Incorrect: An array is a representation and does not inherently impose LIFO service.

C - Correct: A stack is explicitly defined by LIFO, equivalently FILO.

D - Incorrect: A linked list can implement several policies and does not inherently require LIFO.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA01

> An abstract data type (ADT) defines values and operations; representation supplies implementation. Arrays or linked nodes can implement the same LIFO stack. Abstraction keeps relevant details: patient urgency and arrival order matter for service; hair colour does not.

Question sources

Data Structures & Algorithms review questions · Question 1

Question 219 | NS0520

What distinguishes a dynamic data structure from a static one?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Dynamic structures can change size at runtime; static structures have fixed capacity.

DSA02 distinguishes fixed arrays from resizable arrays, and DSA08 allocates linked nodes as needed. Runtime growth or shrinkage is the key distinction; fixed capacity need not mean compile-time allocation.

Primary note proof

Study-note section DSA02

> A fixed array has a chosen capacity; the choice need not occur at compile time. A resizable array allocates a larger block and copies when needed. Inserting in the middle shifts later elements, so it costs linear time. Appending into spare capacity is constant; a resizing append can be linear, though repeated geometric growth gives amortized constant append cost.

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Question sources

Data Structures & Algorithms review questions · Question 2

Question 220 | NS0521

Which classification includes both Trees and Graphs?

A. Linear

B. Homogeneous

C. Non-linear

D. Static

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer C: Non-linear

DSA01 treats arrays, lists, stacks, and queues as linear sequences, trees as hierarchies, and graphs as general connections. Trees and graphs therefore belong to the nonlinear relationship category. Homogeneity describes element types and static describes storage capacity; neither classifies both by their branching relationship structure. C is the appropriate shared category.

Why each choice is right or wrong

A - Incorrect: Linear describes sequence-like structures rather than branching trees and general graphs.

B - Incorrect: Homogeneous concerns uniform element types and does not identify their relationship structure.

C - Correct: Trees and graphs are nonlinear relationship structures, unlike the listed linear sequences.

D - Incorrect: Static concerns storage or capacity decisions, not whether connections branch.

Primary note proof

Study-note section DSA01

> An abstract data type (ADT) defines values and operations; representation supplies implementation. Arrays or linked nodes can implement the same LIFO stack. Abstraction keeps relevant details: patient urgency and arrival order matter for service; hair colour does not.

Question sources

Data Structures & Algorithms review questions · Question 3

Question 221 | NS0522

In an ADT, what two components are always defined?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: An ADT defines its values/data and permitted operations.

DSA01 explicitly defines an abstract data type through values and operations, leaving representation to the implementation. The same LIFO stack can therefore use arrays or linked nodes without changing its contract.

Primary note proof

Study-note section DSA01

> An abstract data type (ADT) defines values and operations; representation supplies implementation. Arrays or linked nodes can implement the same LIFO stack. Abstraction keeps relevant details: patient urgency and arrival order matter for service; hair colour does not.

Question sources

Data Structures & Algorithms review questions · Question 4

Question 222 | NS0523

A structure in C that groups int, char, and float is an example of:

A. Homogeneous

B. Non-homogeneous

C. Linear

D. Static

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer B: Non-homogeneous, or heterogeneous

DSA01 directly identifies a record grouping integer, character, and floating fields as heterogeneous. The different field types distinguish it from an ordinary same-element-type array. Linear describes relationships, and static describes storage behavior, neither of which follows merely from the field types. Thus Non-homogeneous is the requested classification.

Why each choice is right or wrong

A - Incorrect: Homogeneous would mean a uniform element type, contrary to the mixed int, char, and float fields.

B - Correct: The record combines different field types and is therefore heterogeneous or non-homogeneous.

C - Incorrect: Linear concerns relationships or order, not the mixed-type classification asked here.

D - Incorrect: Static concerns allocation or lifetime and is not established merely by these field types.

Primary note proof

Study-note section DSA01

> Arrays, linked lists, stacks and queues are linear. Trees model hierarchies; graphs model general connections. A record grouping integer, character and floating fields is heterogeneous; an ordinary array has one element type.

Question sources

Data Structures & Algorithms review questions · Question 5

Question 223 | NS0524

What does abstraction in algorithms allow programmers to do?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Abstraction preserves relevant behavior and hides irrelevant implementation detail.

DSA01 illustrates abstraction using patient urgency and arrival order while ignoring hair colour. Programmers retain the data and operations needed for the task and separate that contract from representation details. This supports the substantive answer without additional facts.

Primary note proof

Study-note section DSA01

> An abstract data type (ADT) defines values and operations; representation supplies implementation. Arrays or linked nodes can implement the same LIFO stack. Abstraction keeps relevant details: patient urgency and arrival order matter for service; hair colour does not.

Question sources

Data Structures & Algorithms review questions · Question 6

Question 224 | NS0525

Which operation involves combining elements from two sorted sets into one sorted set?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Merging

DSA03 calls combining sequences merging and works through sorted inputs by repeatedly emitting the smaller current front. Advancing only the selected sequence preserves sorted output until the remaining suffix is appended. For mathematical sets, duplicate handling additionally follows the set contract.

Primary note proof

Study-note section DSA03

> Traversal processes every item; searching locates a target; insertion and deletion change membership; updating changes a stored value. Sorting reorders items, while merging combines sequences. To merge sorted [1,4] and [2,3], repeatedly emit the smaller front, producing [1,2,3,4]. Destroying a structure releases owned storage.

Study-note section DSA27

> A set forbids duplicate elements.

Question sources

Data Structures & Algorithms review questions · Question 7

Question 225 | NS0526

What is the term for the operation that releases memory when a data structure is no longer needed?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Destruction or deallocation of owned storage.

DSA03 says destroying a structure releases owned storage. In a linked chain, DSA09 visits and releases each owned node; C++ allocation forms must be matched to their deletion forms, and freed nodes must not be dereferenced.

Primary note proof

Study-note section DSA03

> A C struct groups named fields, including arrays and pointers. A singly linked node needs payload plus a same-type link. C fields cannot have initializers inside the type declaration; modern C++ permits default member initializers. In C++, match new with delete and new[] with delete[]. Repair links before freeing a node; never dereference it afterwards.

Study-note section DSA09

> Deleting the tail of a singly linked list requires its predecessor, even when tail itself is known. Handle empty and single-node lists separately; after removing the only node, set both head and tail to null. To remove a known successor, bypass it before deallocation. Clearing an owned n-node chain visits and releases every node.

Question sources

Data Structures & Algorithms review questions · Question 8

Question 226 | NS0527

Given T(n) = 6n + 4, what is the Big-O complexity?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: O(n), tightly Θ(n).

For n≥1, 6n+4≤10n, while 6n+4≥6n. These constant upper and lower multiples establish linear growth by DSA06. O(n) is the natural tightest-order response, though larger upper bounds also exist.

Primary note proof

Study-note section DSA06

> For eventually nonnegative functions, f(n)=O(g(n)) means f(n)≤c g(n) for some positive c beyond a threshold k. Omega reverses the inequality; Theta requires both. Little-o means f/g tends to zero; little-omega means it grows without bound. These relations can describe best-case, average-case, worst-case or space functions.

Question sources

Data Structures & Algorithms review questions · Question 9

Question 227 | NS0528

Show why f(n) = $3n^2 + 4n + 1$ is $O(n^2)$. What are the values of c and k?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: One valid witness pair is c=8, k=1.

For every n≥1, $4n \le 4n^2$ and $1 \le n^2$, so $3n^2+4n+1 \le 8n^2$. DSA06 supplies the same inequality, proving the required eventual upper bound with c=8 and k=1. Other larger constants may also work.

Primary note proof

Study-note section DSA06

> For eventually nonnegative functions, f(n)=O(g(n)) means f(n)≤c g(n) for some positive c beyond a threshold k. Omega reverses the inequality; Theta requires both. Little-o means f/g tends to zero; little-omega means it grows without bound. These relations can describe best-case, average-case, worst-case or space functions.

Question sources

Data Structures & Algorithms review questions · Question 10

Question 228 | NS0529

Which property of an algorithm ensures it terminates after a finite number of steps?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Finiteness

DSA04 defines finiteness as terminating after finitely many steps for every valid input. It is distinct from definiteness, executable instructions, or correctness: a terminating procedure can still return a wrong answer. The required concept is directly covered.

Primary note proof

Study-note section DSA04

> Finiteness means termination after finitely many steps for every valid input. Definiteness means each instruction has one interpretation; effectiveness means each step is executable. Correctness means meeting the stated result for every valid input. A finite program can still return the wrong answer.

Question sources

Data Structures & Algorithms review questions · Question 11

Question 229 | NS0530

An algorithm has exactly two independent nested loops. Each executes n iterations, and the innermost body performs constant work. Loop-control operations are constant time, with no early exits or hidden nonconstant operations. What is the tight time complexity as n grows?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: Theta(n^2) time.

For each of the n outer iterations, the inner loop executes n times, so the constant-cost body runs $n \times n = n^2$ times. Outer-loop overhead is linear and inner-loop control is also bounded by a constant per body execution. Those contributions do not change the dominant quadratic growth. Under the stated independence, fixed loop count and body-cost assumptions, the tight time complexity is Theta(n^2).

Primary note proof

Study-note section DSA07

> A loop i=1; i<n; i*=2 takes about $\log_2 n$ iterations. Two independent n-iteration loops nested around constant work take Theta(n^2). An n-level ordinary recursion with constant live storage per level needs Theta(n) stack space.

Study-note section DSA05

> Time counts chosen operations; space counts memory, including live recursive frames. Distributed algorithms may also measure communication volume. A single assignment is one unit in the elementary model, not literally one second. Sequential blocks add costs; independent nested loops multiply their iteration counts. Conditional worst-case cost includes the test and the more expensive branch.

Question sources

Data Structures & Algorithms review questions · Question 12

Question 230 | NS0531

Big-Theta (Θ) notation represents:

A. Upper bound only

B. Lower bound only

C. Both upper and lower bound

D. Average case only

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer C: Both upper and lower asymptotic bounds.

DSA06 defines Big-O as an eventual upper bound, Omega as the reverse bound, and Theta as requiring both. Thus Theta expresses a tight growth class up to positive constants. It is independent of choosing best, average, or worst input cases; Average case only confuses the bound relation with an input-distribution choice.

Why each choice is right or wrong

A - Incorrect: An upper bound alone is Big-O and does not supply Theta's lower requirement.

B - Incorrect: A lower bound alone is Omega and does not supply Theta's upper requirement.

C - Correct: Theta requires simultaneous eventual upper and lower constant-factor bounds.

D - Incorrect: Theta can describe any case or resource function, not only an average case.

Primary note proof

Study-note section DSA06

> For eventually nonnegative functions, f(n)=O(g(n)) means f(n)≤c g(n) for some positive c beyond a threshold k. Omega reverses the inequality; Theta requires both. Little-o means f/g tends to zero; little-omega means it grows without bound. These relations can describe best-case, average-case, worst-case or space functions.

Question sources

Data Structures & Algorithms review questions · Question 13

Question 231 | NS0532

Which grows faster asymptotically: O(n log n) or O(n)?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: Representative n log n grows faster than n; Big-O classes themselves overlap.

DSA06 orders representative rates with n before n log n. Their ratio is log n, which grows without bound, so the representative n log n rate grows faster. However an O(n) function is also O(n log n), so membership alone does not compare actual runtimes.

Primary note proof

Study-note section DSA06

> For eventually nonnegative functions, f(n)=O(g(n)) means f(n)≤c g(n) for some positive c beyond a threshold k. Omega reverses the inequality; Theta requires both. Little-o means f/g tends to zero; little-omega means it grows without bound. These relations can describe best-case, average-case, worst-case or space functions.

Study-note section DSA06

> Increasing representative rates: 1, log n, n, n log n, n^2, n^3, 2^n, 3^n, n!, n^n. Logarithm bases differ only by a constant factor.

Question sources

Data Structures & Algorithms review questions · Question 14

Question 232 | NS0533

What is the running time contribution of a single assignment statement according to analysis rules?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: One constant time unit in the elementary-operation model.

DSA05 explicitly treats a single assignment as one unit in an elementary model, not one physical second. The convention abstracts hardware and requires that evaluating the assigned expression itself be constant cost; complicated expressions need their own accounting.

Primary note proof

Study-note section DSA05

> Time counts chosen operations; space counts memory, including live recursive frames. Distributed algorithms may also measure communication volume. A single assignment is one unit in the elementary model, not literally one second. Sequential blocks add costs; independent nested loops multiply their iteration counts. Conditional worst-case cost includes the test and the more expensive branch.

Question sources

Data Structures & Algorithms review questions · Question 15

Question 233 | NS0534

State the Transitivity theorem of Big-O. Why is it useful?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: If f∈O(g) and g∈O(h), then f∈O(h); this composes upper bounds.

DSA06 composes the eventual inequalities f≤c g and g≤d h to obtain f≤cd h beyond both thresholds. Choose the larger threshold and multiply the positive witness constants. This lets a complicated bound pass through a simpler intermediate function.

Primary note proof

Study-note section DSA06

> For n≥1, $3n^2+4n+1 \le 8n^2$: c=8,k=1 witness $O(n^2)$. Constants and lower-order terms disappear from tight polynomial growth. Transitivity multiplies witnesses: f≤c g≤cd h beyond both thresholds.

Question sources

Data Structures & Algorithms review questions · Question 16

Question 234 | NS0535

Rank from slowest to fastest growing: $O(n^2)$, $O(1)$, $O(2^n)$, $O(\log n)$, $O(n)$

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: Increasing representative growth: 1, log n, n, n², 2^n.

DSA06 gives the standard growth ordering, with constants below logarithms, then linear and quadratic functions, then exponential. The ranking compares representative rates, not disjoint Big-O classes, since loose upper bounds overlap.

Primary note proof

Study-note section DSA06

> Increasing representative rates: 1, log n, n, n log n, n^2, n^3, 2^n, 3^n, n!, n^n. Logarithm bases differ only by a constant factor.

Question sources

Data Structures & Algorithms review questions · Question 17

Question 235 | NS0536

What are the three main resources considered in algorithm analysis?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: Time, memory space, and-in distributed settings-communication volume.

DSA05 measures operation counts and live memory, and adds communication volume for distributed algorithms. Thus the third resource is context-dependent rather than a universal resource needed by every local algorithm. These are the three supported categories in the note.

Primary note proof

Study-note section DSA05

> Time counts chosen operations; space counts memory, including live recursive frames. Distributed algorithms may also measure communication volume. A single assignment is one unit in the elementary model, not literally one second. Sequential blocks add costs; independent nested loops multiply their iteration counts. Conditional worst-case cost includes the test and the more expensive branch.

Question sources

Data Structures & Algorithms review questions · Question 18

Question 236 | NS0537

What is the worst-case time complexity of Linear Search and when does it occur?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: Θ(n) worst time, when absent or at the last position.

DSA28 scans until equality or exhaustion. An absent target or one in the final position requires all n comparisons, giving linear worst time; a first-position match instead takes constant time. The operation and cases are directly supported.

Primary note proof

Study-note section DSA28

> Compare the target with successive elements until equal or exhausted. Unsorted data is allowed; sorted data is allowed too. A first-element match costs Theta(1). An absent target or last-position match needs n comparisons, giving Theta(n) worst time. Uniformly likely successful positions average (n+1)/2 comparisons, still linear.

Question sources

Data Structures & Algorithms review questions · Question 19

Question 237 | NS0538

Binary Search requires the list to be:

A. Unsorted

B. Sorted

C. Circular

D. Linked

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer B: Sorted

DSA29 eliminates half an ordered interval according to the midpoint comparison. That elimination is valid only when comparable values follow a consistent order; descending input works with reversed comparisons. Circularity and links are representation properties, not substitutes for sortedness. An unsorted sequence gives no safe smaller-half decision. Therefore Sorted is the required prerequisite.

Why each choice is right or wrong

A - Incorrect: Without order, a midpoint comparison cannot safely eliminate either half.

B - Correct: Consistent sorted order supports the midpoint half-elimination invariant.

C - Incorrect: Circular storage does not establish the value ordering needed by binary search.

D - Incorrect: Linked storage does not imply sortedness and also changes midpoint-access costs.

Primary note proof

Study-note section DSA29

> On an ascending random-access sequence, set lo=0, hi=n-1. Compute mid=lo+(hi-lo)//2. Equality finishes; if target<A[mid], set hi=mid-1; otherwise set lo=mid+1. Stop unsuccessfully when lo>hi. Reverse comparisons for descending order. Comparable keys and valid finite bounds matter; fixed allocation before population is not an extra algorithmic requirement.

Question sources

Data Structures & Algorithms review questions · Question 20

Question 238 | NS0539

How many steps does Binary Search take for a list of 1,000,000 elements (approximately)?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: About 20 worst-case midpoint probes.

DSA29 gives floor($\log_2 n$)+1 probes for n>0. Since 2^19=524288 and 2^20=1048576, one million lies between them, so the worst probe count is twenty under the ordinary interval algorithm. A root/midpoint hit can finish earlier.

Primary note proof

Study-note section DSA29

> Worst probes are floor($\log_2 n$)+1 for n>0: seven for 100 keys, about twenty for one million. Iteration uses constant space; recursion uses logarithmic stack depth. Linked midpoint traversal loses logarithmic total time. Boundary variants retain a candidate rather than stopping at equality.

Question sources

Data Structures & Algorithms review questions · Question 21

Question 239 | NS0540

What is the time complexity of Bubble Sort and why?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: Worst and usual average $\Theta(n^2)$; early-stop best $\Theta(n)$.

DSA32 sums repeated adjacent passes to at most n(n−1)/2 comparisons, yielding quadratic worst and ordinary average work. A no-swap early-stop pass makes sorted input linear, while a fixed pass schedule remains quadratic. The answer needs these implementation and case distinctions.

Primary note proof

Study-note section DSA32

> For ascending order, compare each adjacent pair and swap only when left>right. The largest remaining value moves to the final unsorted position after one left-to-right pass. Shrink the next pass by one position. Quadratic worst/average work comes from up to n(n-1)/2 pair comparisons.

Question sources

Data Structures & Algorithms review questions · Question 22

Question 240 | NS0541

In Selection Sort, what does each pass through the unsorted subarray accomplish?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Select the remaining minimum and put it at the next prefix position.

DSA33 scans the unsorted suffix for its minimum and exchanges it with A[i]. After each meaningful pass, the sorted prefix grows by one and contains the smallest keys placed so far. This differs from inserting the next encountered item into a prefix.

Primary note proof

Study-note section DSA33

> At position i, scan the remaining suffix for its minimum, then exchange it with A[i]. There are n-1 meaningful placements and n(n-1)/2 comparisons regardless of initial order. At most n-1 exchanges of distinct positions are needed. A loop that also swaps the last item with itself can count n swap calls.

Question sources

Data Structures & Algorithms review questions · Question 23

Question 241 | NS0542

How does Insertion Sort differ from Selection Sort conceptually?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Selection finds a suffix minimum; insertion places the next item into a sorted prefix.

DSA33 chooses the globally smallest remaining key for each next output position. DSA34 instead takes the next input item, shifts larger prefix values, and inserts it into their gap. Both grow sorted prefixes, but choose their next item differently.

Primary note proof

Study-note section DSA33

> At position i, scan the remaining suffix for its minimum, then exchange it with A[i]. There are n-1 meaningful placements and n(n-1)/2 comparisons regardless of initial order. At most n-1 exchanges of distinct positions are needed. A loop that also swaps the last item with itself can count n swap calls.

Study-note section DSA34

> Begin with the first element already sorted. For each remaining item, save its value, shift larger preceding values right, and insert it into the gap. That gives n-1 insertion passes. On sorted input each pass needs one ordering check and no shifts: linear time. Reverse order causes 1+2+...+(n-1) shifts: quadratic time.

Question sources

Data Structures & Algorithms review questions · Question 24

Question 242 | NS0543

What is the best-case scenario for Bubble Sort and what is its complexity in that case?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: Already sorted input; Θ(n) with no-swap early stopping.

DSA32 performs one pass over an already sorted array, makes no swaps, and stops when early-exit detection is enabled. That is linear best work. A fixed-pass implementation still executes its quadratic comparison schedule even on the same input.

Primary note proof

Study-note section DSA32

> Early stopping makes a sorted input linear: one no-swap pass finishes. A fixed pass schedule stays quadratic. Strict swaps preserve equal-key order. Cocktail sort runs bubble passes in both directions and is stable with strict swaps.

Question sources

Data Structures & Algorithms review questions · Question 25

Question 243 | NS0544

What are the two components of each node in a singly linked list?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Payload/data and a next-node link.

DSA08 defines a singly linked node as data plus next. The head reference points to the first node, and a null next terminates an ordinary noncircular chain. Head is a list-level reference rather than a second payload field inside every node.

Primary note proof

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Question sources

Data Structures & Algorithms review questions · Question 26

Question 244 | NS0545

How does a circular linked list differ from a singly linked list?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: A circular tail links back to head instead of ending at null.

DSA10 defines circularity by the tail-to-head link and stops traversal after returning to the start. This is independent of singly versus doubly linked: a singly linked list can itself be circular. Compared with the ordinary noncircular singly linked chain, termination changes from null to return-to-start.

Primary note proof

Study-note section DSA10

> Doubly linked nodes add prev; head and tail enable constant-time operations at both ends. Singly linked tail deletion still needs its predecessor. A circular tail links back to head; stop traversal on returning to the start, handling empty separately.

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Question sources

Data Structures & Algorithms review questions · Question 27

Question 245 | NS0546

To delete a node from the front of a singly linked list, what single update is needed?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Advance head to the old head's next link, then reclaim the old node.

DSA09 saves old=head, sets head=head.next, and releases old in an owned C++ chain. The essential link update is advancing head; check empty before dereferencing, and clear tail too if the only node was removed. This answers the operation without hiding ownership work.

Primary note proof

Study-note section DSA09

> Prepend with new.next=head; head=new. To insert after a known predecessor p, set new.next=p.next before p.next=new. Appending with only head requires a tail traversal; maintaining tail avoids it. Delete the head by saving old=head, advancing head=head.next and releasing old in C++.

Study-note section DSA09

> Deleting the tail of a singly linked list requires its predecessor, even when tail itself is known. Handle empty and single-node lists separately; after removing the only node, set both head and tail to null. To remove a known successor, bypass it before deallocation. Clearing an owned n-node chain visits and releases every node.

Question sources

Data Structures & Algorithms review questions · Question 28

Question 246 | NS0547

What extra pointer does each node in a doubly linked list have compared to a singly linked list?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: A prev link to the predecessor.

DSA10 says doubly linked nodes add prev alongside next. That extra link permits backward traversal and supports endpoint updates when head and tail are maintained. It does not automatically give constant arbitrary positional access, which may still need traversal.

Primary note proof

Study-note section DSA10

> Doubly linked nodes add prev; head and tail enable constant-time operations at both ends. Singly linked tail deletion still needs its predecessor. A circular tail links back to head; stop traversal on returning to the start, handling empty separately.

Question sources

Data Structures & Algorithms review questions · Question 29

Question 247 | NS0548

Why do linked lists have slower access times than arrays?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: Arrays compute addresses directly; linked positional access follows a chain.

DSA02 computes an element address from base, index, and element size in constant time. DSA08's chain lacks direct index arithmetic, so reaching an arbitrary position may traverse linearly many links. This comparison concerns positional access rather than every operation: a known-head update can be constant.

Primary note proof

Study-note section DSA02

> For equal-size elements, address(A[i]) = base + i × element_size under zero-based indexing. This arithmetic gives constant-time positional access in the usual RAM cost model. A row-major rectangular matrix uses base + (row × columns + column) × element_size.

Study-note section DSA08

> A singly linked node contains data and next. Head points to the first node; null ends a noncircular chain. Empty means head==null. Nodes allocated as needed grow dynamically, with pointer overhead and no direct index arithmetic.

Question sources

Data Structures & Algorithms review questions · Question 30

Question 248 | NS0549

Convert the infix expression A + B * C - D / E to postfix.

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: ABC*+DE/-

DSA13 works the exact expression. Multiplication groups B*C and division groups D/E before addition and subtraction. Emit A, then B C *, then +, then D E /, then −, producing ABC*+DE/-.

Primary note proof

Study-note section DSA13

> Infix places an operator between operands and uses precedence and parentheses. Prefix places it first; postfix places it last. Their fixed operator arity determines grouping. To convert infix to postfix, output operands immediately and hold operators on a stack. Pop higher-precedence operators before pushing a new operator; also pop equal precedence for left-associative operators. Parentheses temporarily override this order.

Study-note section DSA13

> A+B*C-D/E becomes ABC*+DE/-.

Question sources

Data Structures & Algorithms review questions · Question 31

Question 249 | NS0550

What condition causes stack overflow in an array-implemented stack?

Hint

Use the cited definitions and invariants to distinguish the requested concept from related operations.

Answer: Attempting push when the fixed array is full.

DSA11 distinguishes full storage from the overflow attempt. Under zero-based top-as-last-occupied convention, full means top=C−1; a count convention uses size=C. A push must check this before writing beyond capacity.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA11

> Overflow is an attempted insertion into full storage; underflow is an attempted removal from empty storage.

Question sources

Data Structures & Algorithms review questions · Question 32

Question 250 | NS0551

What are the three main operations of a Stack and their time complexities?

Hint

State the representation and cost model, then count the operations or storage demanded by the question.

Answer: Push, pop, and peek are O(1) for fixed-array or head-linked stacks.

DSA11 defines push insertion, pop removal/return, and peek observation at top. Fixed-array indexing and linked-head updates take constant work. A resizing append or singly linked tail-pop implementation can change the cost, so the guarantee needs the stated representations.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA11

> A count-based alternative uses size=0 and A[size++]=x. Do not combine its empty/full tests with the top-index convention. Fixed-array push/pop are constant time. Linked stacks achieve constant time by inserting and deleting at the head; using a singly linked tail for pop would require a predecessor traversal.

Question sources

Data Structures & Algorithms review questions · Question 33

Question 251 | NS0552

In postfix notation, why are parentheses unnecessary?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Operator position and known arity determine grouping.

DSA13 places postfix operators after their operands. During stack evaluation, each binary operator consumes the two latest operand values in right-then-left order and pushes their result. The sequence therefore encodes grouping without precedence parentheses.

Primary note proof

Study-note section DSA13

> Infix places an operator between operands and uses precedence and parentheses. Prefix places it first; postfix places it last. Their fixed operator arity determines grouping. To convert infix to postfix, output operands immediately and hold operators on a stack. Pop higher-precedence operators before pushing a new operator; also pop equal precedence for left-associative operators. Parentheses temporarily override this order.

Study-note section DSA13

> To evaluate postfix, push operands. At a binary operator, pop right operand first, then left, and push left op right. Build an expression tree for either prefix (preorder) or postfix (postorder); a stack is useful but is not the only possible conversion representation.

Question sources

Data Structures & Algorithms review questions · Question 34

Question 252 | NS0553

What is the difference between a simple array queue and a circular queue?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Circular wraparound reuses freed front slots; a simple noncircular queue may not.

DSA15 distinguishes a rear index reaching the array end from circular modulo arithmetic. A circular queue can insert into positions freed at the front, while a simple noncircular version may report overflow despite those gaps. Capacity and empty/full conventions still must be consistent.

Primary note proof

Study-note section DSA15

> A simple noncircular array queue may report overflow when rear reaches the final index despite released slots at the front. A circular queue reuses slots with modulo arithmetic. Use a count convention: front points to the next removal, size counts items, and insertion slot is (front+size)%C. Empty means size==0; full means size==C.

Question sources

Data Structures & Algorithms review questions · Question 35

Question 253 | NS0554

A stable priority queue serves smaller numeric priority keys first and compares equal priorities by an increasing arrival counter, also smallest first. If two tasks have equal priority, in what order are they served?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: FIFO among equal priorities: the task that arrived earlier is served first.

The priority comparison ties, so the specified comparator proceeds to the arrival counter. An increasing counter gives the earlier arrival the smaller secondary key, causing it to be removed first. Equal-priority tasks are therefore served FIFO under this stable policy. That result follows from the explicitly chosen tie rule; an ordinary priority queue without a stability or comparator rule would not inherently guarantee arrival-order service.

Primary note proof

Study-note section DSA17

> Arrays and lists can also implement priority queues: an unordered sequence gives cheap insertion but linear extremum removal; an ordered sequence reverses that tradeoff. Equal priorities have no inherent FIFO guarantee. If stable service is required, compare (priority, arrival_counter). Choose the priority direction and tie rule before tracing.

Question sources

Data Structures & Algorithms review questions · Question 36

Question 254 | NS0555

What causes queue underflow and what does it indicate?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Dequeue on an empty queue causes underflow.

DSA11 defines underflow as attempted removal from empty storage; DSA15's count convention detects empty using size==0. This indicates that no item exists to return, not that the array is full or that an ordinary emptiness test itself failed.

Primary note proof

Study-note section DSA11

> Overflow is an attempted insertion into full storage; underflow is an attempted removal from empty storage. Merely checking isEmpty does not underflow.

Study-note section DSA15

> A simple noncircular array queue may report overflow when rear reaches the final index despite released slots at the front. A circular queue reuses slots with modulo arithmetic. Use a count convention: front points to the next removal, size counts items, and insertion slot is (front+size)%C. Empty means size==0; full means size==C.

Question sources

Data Structures & Algorithms review questions · Question 37

Question 255 | NS0556

Name three real-world applications of stacks.

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Runtime call frames, undo/redo history, and string or sequence reversal.

DSA12 gives nested calls latest-first returns, moving latest actions between undo and redo stacks, and reversing input by pushing then popping. Each example uses the same latest-pending-item discipline. A new edit after undo normally clears redo history.

Primary note proof

Study-note section DSA12

> Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Question sources

Data Structures & Algorithms review questions · Question 38

Question 256 | NS0557

What property must hold at every node of a Binary Search Tree?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Every left-subtree key is smaller and every right-subtree key larger, for distinct keys.

DSA19 imposes this invariant over entire subtrees at every node, not only immediate children. A descendant can satisfy its parent comparison yet violate an ancestor bound. Duplicate keys require an explicit count or placement policy instead of silently weakening the distinct-key rule.

Primary note proof

Study-note section DSA19

> With distinct keys, every key in a node's left subtree is smaller and every key in its right subtree is larger. Immediate-child comparisons alone are insufficient: root 10, left child 5, and 5's right child 12 violate the root's restriction. Duplicates require a separate count or placement policy.

Question sources

Data Structures & Algorithms review questions · Question 39

Question 257 | NS0558

List the order of nodes visited in Inorder traversal of BST: root=4, left subtree has 1,2,3, right has 5,6,7.

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: 1,2,3,4,5,6,7

DSA20 visits left subtree, node, then right subtree for inorder, and says BST inorder returns sorted keys. The supplied seven distinct keys therefore appear in increasing order even though the internal child shapes are not fully specified. Shape would matter to preorder or postorder.

Primary note proof

Study-note section DSA20

> Preorder visits node,left,right; inorder left,node,right; postorder left,right,node. Apply recursively; null children do nothing. Inorder returns sorted keys only for a BST. Preorder supports copying, postorder supports deleting children before parents, and level order uses a FIFO queue.

Study-note section DSA19

> With distinct keys, every key in a node's left subtree is smaller and every key in its right subtree is larger. Immediate-child comparisons alone are insufficient: root 10, left child 5, and 5's right child 12 violate the root's restriction. Duplicates require a separate count or placement policy.

Question sources

Data Structures & Algorithms review questions · Question 40

Question 258 | NS0559

When deleting a BST node with two children, what are the two valid replacement choices?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Inorder predecessor or inorder successor.

DSA21 identifies the maximum key in the left subtree and minimum key in the right subtree as valid replacement values. Copy one into the deleted node, then remove that value's original node, which has at most one child. These preserve the subtree order invariant.

Primary note proof

Study-note section DSA21

> A leaf can be detached by clearing its parent link. For one child, make the parent bypass the node and point to the child. For two children, copy the inorder predecessor (left-subtree maximum) or successor (right-subtree minimum), then delete that replacement from its original location. That second deletion has at most one child.

Question sources

Data Structures & Algorithms review questions · Question 41

Question 259 | NS0560

What is the time complexity of BST search in the best case vs worst case?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Best successful search Θ(1); arbitrary BST worst Θ(n), balanced worst Θ(log n).

DSA19 gives a constant root hit and O(height) search. A chain has linear height, producing linear worst work; logarithmic height disciplines in DSA22 improve that worst bound. Calling every BST logarithmic would incorrectly assume balance.

Primary note proof

Study-note section DSA19

> Search compares at each node and follows one child until equality or null. Insertion uses that same path to attach a new leaf. Operations cost O(height): logarithmic for balanced trees, linear for a chain. Best-case successful search is Theta(1), a root hit. An optimal BST minimizes expected search cost using access probabilities; constructing it is a different problem from searching it.

Study-note section DSA22

> AVL balance factor is height(left)-height(right); every node must have factor -1,0 or 1. After insertion or deletion, update heights upward and rotate where necessary. LL imbalance needs a right rotation; RR needs left; LR rotates the left child left then the node right; RL is symmetric. Rotations preserve inorder key order.

Question sources

Data Structures & Algorithms review questions · Question 42

Question 260 | NS0561

What is the key difference between BFS and DFS graph traversal?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: BFS explores distance layers with FIFO; DFS finishes branches with a stack or recursion.

DSA41 enqueues discoveries in layer order, while DSA42 resumes the latest unfinished branch before later neighbors. Both can traverse all reachable vertices, but their pending-work discipline and visitation order differ. BFS also has the unweighted minimum-edge guarantee.

Primary note proof

Study-note section DSA41

> Mark the source when enqueuing it. Repeatedly dequeue a vertex, examine its neighbours, and enqueue each unvisited neighbour while marking it. Store predecessor[v] when discovering v if a path must be reconstructed. FIFO ensures vertices at distance d are explored before later distance layers.

Study-note section DSA42

> Mark a vertex, then recursively visit each unmarked neighbour before continuing with later neighbours. An explicit stack can store pending work instead. Both implement latest-pending-work first; a queue would change that discipline. Marking prevents infinite repetition around graph cycles.

Question sources

Data Structures & Algorithms review questions · Question 43

Question 261 | NS0562

A directed graph has edges with direction; an undirected graph does not. Give one real-world example of each.

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Directed: following accounts; undirected: a mutual friendship relationship.

DSA40 gives following as directed because u following v need not imply v following u. It gives mutual friendship as undirected, representing a symmetric connection. The examples concern relation semantics rather than whether lines happen to be drawn with arrows.

Primary note proof

Study-note section DSA40

> A graph has vertices V and edges E. Directed edges distinguish u→v from v→u; undirected edges represent two-way relationships. Following accounts is directed; a mutual friendship relation is undirected. Weighted edges carry costs; unweighted shortest-path questions count edges. DAG expands to directed acyclic graph. Trees are connected acyclic graphs, although graph data structures may also contain cycles.

Question sources

Data Structures & Algorithms review questions · Question 44

Question 262 | NS0563

Why is a dense graph more computationally expensive to process than a sparse graph?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Edge-processing algorithms do more work when E is larger.

DSA40 says dense simple graphs approach quadratic edge counts and adjacency lists use Θ(V+E) space. Traversals or algorithms inspecting every edge therefore require more work as density increases. This is not universal: a vertex-only operation need not inspect edges.

Primary note proof

Study-note section DSA40

> A matrix uses Theta(V^2) space and constant edge lookup. Adjacency lists use Theta(V+E) space and enumerate neighbours. Dense simple graphs approach quadratic edge counts. Examining every edge costs more on denser graphs; vertex-only operations need not inspect every edge.

Study-note section DSA41

> BFS finds minimum-edge paths in unweighted graphs, not arbitrary minimum-weight paths. With adjacency lists it costs O(V+E); scanning a matrix costs O(V^2). To traverse a disconnected graph, restart from each unvisited vertex. Neighbour order affects tied traversal order but not minimum distances.

Question sources

Data Structures & Algorithms review questions · Question 45

Question 263 | NS0564

In a full binary tree, every node has either 0 or 2 children. How does this differ from a complete binary tree?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Full restricts each node to zero or two children; complete restricts level filling.

DSA18 defines full by each node's child count and complete by filled earlier levels plus left-filled final level. Those are distinct conditions: completeness can allow one child at the boundary, while fullness alone need not keep every level filled.

Primary note proof

Study-note section DSA18

> Binary means at most two children. Full means each node has zero or two children. Complete means all earlier levels are filled and the last fills from left. Perfect means all leaves share a level and every internal node has two children. Balanced means height stays logarithmic under a stated rule; it does not require a perfect shape.

Question sources

Data Structures & Algorithms review questions · Question 46

Question 264 | NS0565

What distinguishes a weighted graph from an unweighted graph, and when would you prefer weighted?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Weighted edges carry differing costs; use them when edge costs matter.

DSA40 assigns costs to weighted edges and says unweighted shortest-path questions count edges. Distances or travel costs can therefore justify weights when fewer edges is not the true objective. DSA43 then distinguishes weighted shortest paths from unweighted BFS.

Primary note proof

Study-note section DSA40

> A graph has vertices V and edges E. Directed edges distinguish u→v from v→u; undirected edges represent two-way relationships. Following accounts is directed; a mutual friendship relation is undirected. Weighted edges carry costs; unweighted shortest-path questions count edges. DAG expands to directed acyclic graph. Trees are connected acyclic graphs, although graph data structures may also contain cycles.

Study-note section DSA43

> For nonnegative weights, standard Dijkstra initializes source distance zero and others infinity, repeatedly finalizes the unsettled vertex with smallest distance, and relaxes each outgoing edge: dist[v]=min(dist[v],dist[u]+w(u,v)). A min-priority queue supports the next choice. BFS is enough when every edge has equal cost; negative edges require another justified method, such as Bellman-Ford.

Question sources

Data Structures & Algorithms review questions · Question 47

Question 265 | NS0566

If a BST receives keys in sorted order (1,2,3,4,5), what shape does the tree take and what is the search complexity?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: An ordinary unbalanced insertion forms a right chain; worst search is Θ(n).

DSA19 explicitly traces sorted insertion 1,2,3,4,5 into a right chain. Each new larger key follows right links to the tail, making the height linear. Worst search follows that chain, while a root hit remains constant. Balanced insertion would rotate and is a different assumption.

Primary note proof

Study-note section DSA19

> Search compares at each node and follows one child until equality or null. Insertion uses that same path to attach a new leaf. Operations cost O(height): logarithmic for balanced trees, linear for a chain. Best-case successful search is Theta(1), a root hit. An optimal BST minimizes expected search cost using access probabilities; constructing it is a different problem from searching it.

Study-note section DSA19

> Sorted insertions 1,2,3,4,5 instead form a right chain.

Question sources

Data Structures & Algorithms review questions · Question 48

Question 266 | NS0567

When would you choose a linked list over an array? Give a specific scenario.

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Choose links for frequent known-location updates, such as a doubly linked history cursor.

DSA10 gives browser history with a doubly linked cursor as an application. With references to current insertion or removal positions, link changes avoid shifting an array's later elements; arbitrary positional lookup can still be linear. This concrete scenario supports the choice without claiming universal superiority.

Primary note proof

Study-note section DSA10

> Lists suit changing sequences when references to insertion positions are already available. Finding a position is still linear. Lists can hold polynomial terms (coefficient, exponent), directory or block links, child lists in nonbinary trees, and hash chains. Browser history can use a doubly linked cursor. A playlist may be circular; airline seat lookup requires a specified seat-record model, not an automatic linked-list answer.

Study-note section DSA02

> A fixed array has a chosen capacity; the choice need not occur at compile time. A resizable array allocates a larger block and copies when needed. Inserting in the middle shifts later elements, so it costs linear time. Appending into spare capacity is constant; a resizing append can be linear, though repeated geometric growth gives amortized constant append cost.

Question sources

Data Structures & Algorithms review questions · Question 1

Question 267 | NS0568

How does a stack differ from a queue? Give an application for each.

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Stack: LIFO for undo; queue: FIFO for equal-priority printer jobs.

DSA11 gives latest-first stack service and DSA12 applies it to undoing the latest edit. DSA14 gives arrival-order queues for equal-priority printing. Both ADTs may use arrays or nodes; their service order, not that representation, is the distinction.

Primary note proof

Study-note section DSA11

> Stack policy is LIFO, equivalently FILO. Push inserts, pop removes/returns top, and peek only observes. With zero-based occupied slots, top=-1 is empty and top=C-1 is full. Check before A[++top]=x or returning A[top--].

Study-note section DSA14

> FIFO suits equal-priority printer jobs, shared-resource waiting, asynchronous buffers and level-order exploration. A play sequence can also use a queue. Distinguish airline seat records from waiting requests; an application name alone does not force one representation.

Study-note section DSA12

> Undo moves the latest action from undo to redo; redo moves it back.

Question sources

Data Structures & Algorithms review questions · Question 2

Question 268 | NS0569

Why is a balanced BST (like AVL) preferred over a regular BST? Explain the worst-case scenario.

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Balance prevents linear-height chains and guarantees logarithmic worst search.

DSA22 maintains AVL balance factors through rotations that preserve inorder. DSA19 shows ordinary sorted insertion producing a chain, where a search can traverse all nodes. Maintaining logarithmic height prevents that worst degradation, though root-hit best time remains constant.

Primary note proof

Study-note section DSA22

> AVL balance factor is height(left)-height(right); every node must have factor -1,0 or 1. After insertion or deletion, update heights upward and rotate where necessary. LL imbalance needs a right rotation; RR needs left; LR rotates the left child left then the node right; RL is symmetric. Rotations preserve inorder key order.

Study-note section DSA19

> Search compares at each node and follows one child until equality or null. Insertion uses that same path to attach a new leaf. Operations cost O(height): logarithmic for balanced trees, linear for a chain. Best-case successful search is Theta(1), a root hit. An optimal BST minimizes expected search cost using access probabilities; constructing it is a different problem from searching it.

Question sources

Data Structures & Algorithms review questions · Question 3

Question 269 | NS0570

What is a hash collision? Name two methods to resolve collisions and compare them.

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: A collision maps different keys to one home address; compare chaining and open addressing.

DSA26 stores colliding entries in bucket lists for chaining, whereas open addressing probes other array slots. Chaining uses link or bucket overhead and expected O(1+α) work; probing keeps entries inside the table and requires careful load and deletion policies. Neither prevents all collisions.

Primary note proof

Study-note section DSA26

> Separate chaining stores colliding entries in a bucket list. Load alpha=n/m is average chain length; expected operations are O(1+alpha) under suitable distribution assumptions, with O(n) worst cases. Open addressing keeps entries inside the array and probes until a key or an available slot is found.

Study-note section DSA26

> Linear probing tries (h(k)+j)%m. For size 7 and keys 10,17,24, all home addresses are 3 and occupied positions become 3,4,5. Quadratic probing changes the step; double hashing uses h1(k)+j h2(k), modulo m, with a step relatively prime to m to cover all slots. Neither method guarantees no collisions. Deletion needs tombstones or reinsertion so searches do not stop prematurely.

Question sources

Data Structures & Algorithms review questions · Question 4

Question 270 | NS0571

What is the difference between BFS and DFS? Which finds the shortest path?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: BFS uses FIFO layers and finds shortest unweighted paths; DFS uses branch-first stack order.

DSA41 guarantees minimum-edge paths for BFS on unweighted graphs. DSA42 says DFS paths are reachability witnesses but need not be shortest. Differing edge weights require an appropriate weighted-path algorithm, such as Dijkstra under nonnegative weights.

Primary note proof

Study-note section DSA41

> BFS finds minimum-edge paths in unweighted graphs, not arbitrary minimum-weight paths. With adjacency lists it costs O(V+E); scanning a matrix costs $O(V^2)$. To traverse a disconnected graph, restart from each unvisited vertex. Neighbour order affects tied traversal order but not minimum distances.

Study-note section DSA42

> Using the same graph and alphabetical neighbours, recursive DFS can discover A,B,D,C. Its path is a valid reachability witness but not necessarily shortest. Adjacency-list cost is O(V+E), with O(V) visited state and possible stack depth. DFS supports cycle and connectivity analysis; a directed cycle test distinguishes active recursion vertices from already completed ones.

Study-note section DSA43

> BFS is enough when every edge has equal cost; negative edges require another justified method, such as Bellman-Ford.

Question sources

Data Structures & Algorithms review questions · Question 5

Algorithm Design & Analysis review questions · Question 4

Question 271 | NS0572

A text editor needs to implement multi-level undo. Which data structure is most appropriate and why?

Hint

Use the cited invariant or operation rule, state its assumptions, and trace a small case to justify the result.

Answer: Use an undo stack, with a second redo stack if needed.

DSA12 moves the latest action from undo to redo and reverses that movement for redo. Pushing each new edit and popping for undo naturally matches the most-recent-edit-first requirement. A new edit after undo normally clears redo history.

Primary note proof

Study-note section DSA12

> Calls nest on a runtime stack; the latest call returns first. Bracket matching pushes opening brackets and matches each closing bracket against top. Reversal pushes input then pops it. Undo moves the latest action from undo to redo; redo moves it back. A new edit after undo normally clears redo. DFS and backtracking use the same latest-pending-choice discipline.

Question sources

Data Structures & Algorithms review questions · Question 6

Web Programming

Questions & Worked Answers

305 questions + 1 source variants

Question 1 | Q00008

Which symbol is used to write a single line comment in JavaScript?

A. <?

B. <!---------->

C. //

D. &&

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer C: //

A single-line JavaScript comment starts with two forward slashes (p.46); text following it on that line is ignored. Evidence: PDF p. 46.

Why each choice is right or wrong

A - Incorrect: PHP-style opening delimiter, not the stated JS comment form.

B - Incorrect: HTML-style comment delimiters do not match the JS rule.

C - Correct: Exactly the double forward slash specified on p.46.

D - Incorrect: Logical conjunction syntax is not the double-slash comment delimiter.

Primary note proof

Study-note section WEB15

> // comments end at line end; /* */ span a block.

Study-note section WEB23

> Use <?php ...

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 46 | lines 4-10

> 4.3 JavaScript Comments
> As with HTML and CSS, it is possible to write comments into a JavaScript code
> that will be ignored by the browser, and exist to provide instructions to fellow

developers on how the code works. There are two types of JavaScript comments
- A single line comment written after a double forward slash (//)
// I am a single line comment
- A multiline comment is written between the strings (/* */)

Question sources

MOE Computer Science Exit Exam · 2016 · Question 8

Question 2 | Q00018

Which attribute of INPUT tag defines the number of characters that can be displayed in a text box without scrolling?

A. SIZE

B. MAXLENGTH

C. VALUE

D. TYPE

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer A: SIZE controls approximate visible text-input width.

WEB09 distinguishes size from maxlength: size describes the approximate visible width of a text input, while maxlength limits how much text can be entered. That matches the stem's focus on characters displayed without scrolling. VALUE provides initial content and TYPE selects the control kind; neither is the width-setting attribute requested here.

Why each choice is right or wrong

A - Correct: SIZE controls approximate visible width for a text input, matching the display-without-scrolling description.

B - Incorrect: MAXLENGTH limits entered text length rather than determining how wide the input appears.

C - Incorrect: VALUE supplies the control's content, not its visible character-width setting.

D - Incorrect: TYPE selects the input-control kind rather than its approximate visible width.

Primary note proof

Study-note section WEB09

For a text input, size indicates approximate visible width, whereas maxlength limits entered text length.

Study-note section WEB-S02

For a text input, the value attribute supplies its initial text value; it does not specify the control's width.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 22 | lines 2-9

1. Text inputs create using the <input> tag, an input control can take many forms
using the type attribute. Possible values for type attribute
- Text - most common used for text inputs
- Password - for password fields
- Checkbox
- Radio
Other attributes for <input> tag
- Name, value, size, maxlength

Question sources

MOE Computer Science Exit Exam · 2016 · Question 18

Question 3 | Q00020

Which method can be used to divide a browser into smaller sub windows so that each sub window displaying a different HTML document?

A. Using paragraphs

B. Using frames

C. Using images

D. Using tables

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer B: Using frames

Frames display independent windows/subwindows and each frame may contain a different web page (p.20). This matches dividing the browser into document views. Evidence: PDF p. 20, PDF p. 20, PDF p. 17.

Why each choice is right or wrong

A - Incorrect: Paragraphs structure text (p.15), not separate document views.

B - Correct: Frames explicitly provide the required subwindows.

C - Incorrect: Images embed visual content, not independent HTML documents.

D - Incorrect: Tables structure rows and columns, not separate document views.

Primary note proof

Study-note section WEB10

Legacy frameset/frame divides a window into separate documents; frameset replaced body in old frameset documents.

Study-note section WEB06

<p> marks paragraphs;
 is a line break and <hr> a thematic break.

Study-note section WEB08

<table> contains rows <tr>, header cells <th> and data cells <td>; a td is not the table itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 20 | lines 4-10

2.2.9 HTML Frames
HTML frames allows us to present documents in multiple views, which may be independent windows or sub-windows. Multiple views offer designers a way to keep certain information visible, while other views are scrolled or replaced.
An HTML document that describes frame layout has a different markup than HTML document without frames. A standard document has one HEAD section and one BODY. A frameset document has a HEAD and a FRAMESET in place of BODY.

Web Programming module | PDF page 20 | lines 24-28

2.2.9.2 Internal Frame
The Frame tag define the particular area within an HTML file where another HTML web page can be displayed. A <frame> tag is used with in a <frameset>, and it divides a webpage into multiple sections or frames, and each frame can contain different web pages.

Web Programming module | PDF page 17 | lines 23-29

2.2.7 HTML Table
A very common task in html is structuring tabular data. HTML has a number of elements and attributes just for this purpose. This section will explain how to create tables using the different elements provided by html.
A table is a structured data made up of rows and columns. To create table in html use the fowling elements table, tr, th, td, and captions
tr stands for table row, th stands for table header and td stands for table data

Question sources

MOE Computer Science Exit Exam · 2016 · Question 20

Question 4 | Q00037

Which statement is false about potential benefits of well-designed external style sheets? Treat benefits as possibilities, not guarantees.

A. Style sheets can support accessible presentation

B. Style sheets can create a consistent appearance

C. Style sheets can make HTML pages easier to maintain

D. Style sheets cannot reduce repeated markup or transferred file size

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer D: Style sheets cannot reduce repeated markup or transferred file size

External CSS lets several pages reuse presentation rules, making visual changes easier to maintain and helping consistency. Reuse and caching can reduce repeated markup and transfers, although savings depend on the implementation. Accessible presentation is also possible, but CSS alone does not establish accessibility. The false statement is the absolute denial of any size or transfer savings; the other choices describe potential benefits rather than universal promises.

Why each choice is right or wrong

A - Incorrect: Accessible presentation can be supported, though semantic markup and usable controls are still needed.

B - Incorrect: Reused rules can give pages a consistent appearance.

C - Incorrect: Central rules reduce repeated presentation edits across pages.

D - Correct: This absolute denial is false: reuse and caching can reduce repeated markup and transfers.

Primary note proof

Study-note section WEB11

> External CSS promotes reuse, consistent design and caching; it does not guarantee that every page becomes smaller or accessible.

Study-note section WEB07

> Give controls programmatic labels, clear link text and keyboard access; visual styling alone does not establish accessibility.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 28 | lines 27-31

> 3. External style sheets
> An external style sheet is used when the style is applied to many pages. With an
> external style sheet, you can change the look of an entire website by changing one file

(the style sheet). Each page should be linked to the style sheet using the <link> tag inside the head section.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 37

Question 5 | Q00062

What will be the out of the following JavaScript statement? Math.abs(-9)

A. 9

B. -9

C. 1

D. 0

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer A: 9

WEB18 explicitly evaluates Math.abs(-9) as 9. Absolute value removes the negative sign from a negative numeric argument and leaves its magnitude, so the result is neither the original negative number nor a Boolean-like one or zero. This is an expression result; no rounding or text conversion is involved.

Why each choice is right or wrong

A - Correct: The absolute value of negative nine is its nonnegative magnitude, nine.

B - Incorrect: Negative nine is the input; absolute value does not preserve its negative sign.

C - Incorrect: Math.abs returns numeric magnitude rather than a Boolean-like value of one.

D - Incorrect: The input is nonzero, so its absolute value is not zero.

Primary note proof

Study-note section WEB-S08

For finite numeric arguments, Math.abs(x) gives the nonnegative magnitude, Math.floor(x) gives the greatest integer no larger than x, and Math.sqrt(x) gives the nonnegative square root for nonnegative x.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 84 | lines 17-21

5.16 PHP Mathematical Functions
The math functions can handle values within the range of integer and float types.The PHP math functions are part of the PHP core. No installation is required to use these functions.
Here are some PHP math functions, you can get the whole list from the internet
abs - Absolute value

Web Programming module | PDF page 85 | lines 6-11

Example
<?php
echo abs(-4.2); // 4.2 (double/float)
echo abs(5); // 5 (integer)
echo abs(-5); // 5 (integer)
?>

Question sources

MOE Computer Science Exit Exam · 2016 · Question 62

Question 6 | Q00064

In the legacy browser expression document.bgColor, what is bgColor?

A. bgColor is a method

B. bgColor is the document object

C. bgColor is a property

D. bgColor is an event handler

Hint

Distinguish the browser object or dialog role from the kind of response it collects.

Answer C: bgColor is a property

The expression uses the lowercase document object name and the exact camelCase member bgColor. In this legacy API, bgColor is a property representing background color, rather than a method call or event handler. Asking specifically about the member avoids confusing it with window.document, which is itself a property. Current code normally changes the background through document.body.style.backgroundColor rather than this legacy interface.

Why each choice is right or wrong

A - Incorrect: No function is invoked or named by this legacy color member.

B - Incorrect: document is the object on which the member is accessed, not the bgColor value.

C - Correct: The legacy member stores a background-color setting.

D - Incorrect: The member is a color property, not an event callback.

Primary note proof

Study-note section WEB22

> document.bgColor is a legacy property; Document.bgcolor has wrong object/case.

Study-note section WEB22

> In a normal browser page, window exposes the browsing context and document is its document property.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 53 | lines 2-3

> Example
> element.style.backgroundColor = “red”;

Web Programming module | PDF page 56 | lines 17-25

> All global JavaScript objects, functions, and variables automatically become
> members of the window object.
> Global variables are properties of the window object
> Global functions are methods of the window object.
> Even the document object (of the HTML DOM) is a property of the window
> object:
> window.document.getElementById("header");
> is the same as:
> document.getElementById("header");

Question sources

MOE Computer Science Exit Exam · 2016 · Question 64

Question 7 | Q00069

Look at the following valid PHP array:
$list = array("outside" => "red", "inside" => "brown", "top" => "black", "under" => "blue");
Which exact string key indexes the value "black"?

A. brown

B. outside

C. under

D. top

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: top

PHP associative arrays map explicit keys to values. In each pair, the text before the contiguous => operator is the key and the text after it is the value. The pair containing black therefore uses the key top. String-key case matters, so the lowercase spelling is retained in the corrected choices. The other offered strings either identify another entry or are themselves a value rather than the requested key.

Why each choice is right or wrong

A - Incorrect: brown is the value associated with inside, not the key of black.

B - Incorrect: outside identifies red.

C - Incorrect: under identifies blue.

D - Correct: top is the exact key paired with black.

Primary note proof

Study-note section WEB26

> PHP arrays map integer/string keys to values.

Study-note section WEB26

> Associative syntax uses a contiguous => operator: ["top"=>"black"] stores black under key"top".

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 69 | lines 4-22

> Syntax
> An array can be created using the array() language construct. It takes any number
> of comma-separated key => value pairs as arguments.
> array(
> key => value,
> key2 => value2,
> key3 => value3,
> ...
>)
> E.g.
> <?php
> $array1 = array("Item 1", "Item 2", "Item 3", "Item 4");
> $array2 = array('item1' => "Item 1",
> 'item2' => "Item 2",
> 'item3' => "Item 3",
> 'item4' => "Item 4");
> ?>
> The use of => is similar to the regular = assignment operator, except that you are
> assigning a value to an index and not to a variable.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 69

Question 8 | Q00075

Which statement is false about modern JavaScript running in a browser sandbox?

A. The language cannot support sophisticated applications

B. Scripts can access selected client devices through permission-based APIs

C. Scripts have restricted access to arbitrary client files and hardware

D. Script source can be supplied as text and executed by the browser

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer A: The language cannot support sophisticated applications

Browser JavaScript supports sophisticated applications; its sandbox is a restriction on access to resources, not proof of an inherently weak language. Selected file or device access can be exposed through permission-based APIs, while arbitrary hardware access remains restricted. JavaScript source can be included in pages or loaded from a script file. Thus the language-capability claim is false, while the qualified environment and source-delivery claims are compatible.

Why each choice is right or wrong

A - Correct: The note explicitly allows sophisticated applications, so this denial is false.

B - Incorrect: Permission-based APIs allow selected device access.

C - Incorrect: Sandbox restrictions limit arbitrary resource access.

D - Incorrect: Script inclusion supplies source for browser execution.

Primary note proof

Study-note section WEB01

> Browser scripts are sandboxed, but may access selected files/devices through permission-based APIs.

Study-note section WEB01

> They are not inherently limited to trivial programs, nor can they freely access arbitrary client hardware.

Study-note section WEB15

> Use <script src="app.js"></script> for an external .js file; script content has no HTML closing-tag syntax inside its JavaScript source.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 3-7

> 4.1 Introduction to JavaScript
> JavaScript (JS) is a lightweight, interpreted or compiled programming language. It is a single threaded, dynamic language, supporting object oriented, functional programming styles. Do not confuse JavaScript with the Java Programming language. The two-programming language have very different syntax, semantics, and use.

Web Programming module | PDF page 44 | lines 21-28

> We can use JavaScript in coordination with Application programming interfaces (APIs). APIs provide us extra information. They are ready made code building blocks that are used to simplify programming in JavaScript. There are two types of APIs
> - Third party API → are not built into the browser. We have to grab the code from the web. Examples: Twitter API, Google maps API etc.
> - Browser API → are built into the browser. Examples DOM (Document object model) which is covered later in this chapter, Geolocation API, Canvas and WebGL, Audio and Video elements

Question sources

MOE Computer Science Exit Exam · 2016 · Question 75

Question 9 | Q00155

Which one of the following is false?

A. A web browser is a software that runs on a server

B. A user agent renders resources for a user to view

C. A website is a collection of resources in various forms

D. A web page is the basic unit of information storage on the www

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: A web browser is a software that runs on a server

The module places web pages on servers and says web clients view them using a browser (p.9). Thus A reverses the browser role in the ordinary client/server description. Web pages hold web information (p.8); B and C describe the compatible rendering/resource roles, though they are not verbatim definitions. Evidence: PDF p. 9, PDF p. 8, PDF p. 9.

Why each choice is right or wrong

A - Correct: Client-side browser role contradicts this server-role claim in the teaching model.

B - Incorrect: A browser views pages, consistent with rendering resources.

C - Incorrect: Website files are hosted together (p.9), consistent with this collection description.

D - Incorrect: Page 8 explicitly says web information is stored in Web pages.

Primary note proof

Study-note section WEB01

> A browser/user agent requests resources and renders HTML/CSS, then runs permitted JavaScript.

Study-note section WEB01

> A web/HTTP server receives requests and returns representations, sometimes invoking application code.

Study-note section WEB02

> A website groups resources; a page is one document/resource.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 9 | lines 2-4

> • Web pages are files stored on computers called Web servers.
> • Computers reading the Web pages are called Web clients.
> • Web clients view the pages with a program called a Web browser.

Web Programming module | PDF page 8 | lines 29-34

> • The World Wide Web (WWW) is most often called the Web.
> • The Web is a network of computers all over the world.
> • All the computers in the Web can communicate with each other.
> • All the computers use a communication standard called HTTP.
> How does the WWW work?
> • Web information is stored in documents called Web pages.

Web Programming module | PDF page 9 | lines 13-21

> Web hosting is the place where all the files of your website live. Web hosting is a service that allows organizations and individuals to post a website or a web page onto the Internet. A web host, or web hosting service provider, is a business that provides the technologies and services needed for the website or webpage to be viewed in the Internet. Websites are hosted, or stored, on special computers called servers. When Internet users want to view your website, all they need to do is type your website address or domain (the domain name for the site has to be registered) into their browser. Their computer will then connect to your server and your webpages will be delivered to them through the browser.

Question sources

Addis Ababa University mock exam · Question 57

Question 10 | Q00157

Which one of the following is the correct syntax for document type declaration in HTML5?

A. </doctype html>

B. <!doctype! Html>

C. <doctype! Html>

D. <!doctype html>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer D: <!doctype html>

The module examples begin <!DOCTYPE html> (p.21 and p.30). D has that declaration form, with lowercase keyword; the other choices change its punctuation. Evidence: PDF p. 21, PDF p. 30.

Why each choice is right or wrong

A - Incorrect: A closing-tag slash is not the demonstrated declaration.

B - Incorrect: Extra exclamation after doctype is absent from the example.

C - Incorrect: Missing exclamation immediately after <.

D - Correct: Matches the module declaration structure.

Primary note proof

Study-note section WEB06

> <!DOCTYPE html> selects modern standards mode; <html> is the document root, <head> holds metadata, and <body> contains document content.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 21 | lines 2-3

```
<!DOCTYPE html>
<html>
```

Web Programming module | PDF page 30 | lines 21-22

```
<!DOCTYPE html>
<html lang="en">
```

Question sources

Addis Ababa University mock exam · Question 59

Computer Science Exit Exam · Sene 2016 · Question 16-1

Computer Science Exit Exam practice collection · Question 38

Question 11 | Q00158

In HTML, how do we create a link to another site?

A. click here

B. click here

C. click here

D. <a "http://www.somedomain.com">click here

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer A: click here

Page 14 requires an anchor with an href containing the web address. A supplies that attribute and encloses link text between a and /a. Evidence: PDF p. 14.

Why each choice is right or wrong

A - Correct: Exactly the anchor/href pattern on p.14.

B - Incorrect: url does not substitute for the stated href attribute.

C - Incorrect: link does not substitute for href.

D - Incorrect: A bare quoted URL omits the required href attribute.

Primary note proof

Study-note section WEB07

> Use Read for a link; href is the destination, not url/link.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 14 | lines 5-9

> In HTML, a link is created by wrapping the text or other content inside an <a> tag (anchor tag) and using the href attribute, also known as Hypertext reference, or target that contains the web address.
> Here is an example of a link to the google website
> Google.

Question sources

Addis Ababa University mock exam · Question 60

Question 12 | Q00159

Which one of the following form attribute is used to assign the server-side script file that processes the form data?

A. Action

B. Autocomplete

C. Method

D. Target

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer A: Action

The form attribute action points to the backend script processing collected data. Method instead specifies upload method; target controls the result window.

Why each choice is right or wrong

A - Correct: Backend processing script is precisely action.

B - Incorrect: Autocomplete is not assigned the processing-script role.

C - Incorrect: Method determines the submission method, not script filename.

D - Incorrect: Target determines window/frame for the result.

Primary note proof

Study-note section WEB09

> A form’s action identifies its processing URL, method its submission method, and enctype its encoding.

Study-note section WEB-S02

> A form’s target identifies the window or frame in which the submitted result is displayed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 21 | lines 23-31

> Form attributes:
> - action → Backend script ready to process the data collected
> - method → method used to upload Data. The most frequently used
> - target → Specify the target window or frame where the result of the script will be displayed. It takes values like _blank, _self, _parent etc.
> - enctype → tells how the browser encodes data. Possible values
> 1. application/x-www-form-urlencoded → most common and used from simple scenarios
> 2. mutlipart/form-data → used to upload binary data

Question sources

Addis Ababa University mock exam · Question 61

Computer Science Exit Exam · Sene 2016 · Question 1-2

Computer Science Exit Exam · Sene 2016 · Question 31-4

Computer Science Exit Exam practice collection · Question 39

Question 13 | Q00160

What does this JavaScript print in an ordinary browser page?
<script>
let x = 4 + "4";
document.write(x);
</script>

A. 44

B. 8

C. 4

D. The code contains an error

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer A: 44

The addition expression is complete and ends with a semicolon. Because one operand is a string, + performs concatenation here: the numeric operand contributes its text and the result is the string 44. document.write then emits those two digits.

Why each choice is right or wrong

A - Correct: The string operand makes the two digit representations concatenate.

B - Incorrect: Numeric addition would require numeric operands rather than the given string.

C - Incorrect: The right operand contributes another digit to the output.

D - Incorrect: The expression is syntactically valid.

Primary note proof

Study-note section WEB16

> + can concatenate when a string is involved; arithmetic / converts ordinary numeric strings.

Study-note section WEB22

> document.write is a method/function property; writing after document load can replace the document.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 47 | lines 21-28

- Number → represents integers and floating points. There are many operations we can perform on a number multiplication *, division /, addition +, subtraction - and so on
- String → a string in JavaScript must be surrounded by quotes. We have three different quotes
1. Single quotes --- ‘single’
2. Double quotes --- ‘double’
3. Backticks -- `hello`. In a standard US keyboard, you can find the

Web Programming module | PDF page 48 | lines 23-33

4.5.3 Data Type Conversion
Type conversations are a very common occurrence in JavaScript. Operators and functions converts the values given to them to the right type. Mathematical operations convert value to numbers.
- String conversion → we can use String(value) function to convert a value to string
- Numeric Conversion → happens automatically in mathematical operations and expression. We can use the Number(value) function to explicitly convert a value to a number:
let str = "123";
let num = Number(str);

Question sources

Addis Ababa University mock exam · Question 62

Question 14 | Q00161

What does the following valid PHP code output?

```
<?php
$color = "maroon";
$var = $color[2];
echo "$var";
?>
```

A. a

B. Error

C. var

D. r

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer D: r

PHP string offsets start at zero. The characters of maroon at offsets zero, one and two are m, a and r, so the selected character is r. The variable is interpolated by the double-quoted echo string and no extra spacing is added.

Why each choice is right or wrong

A - Incorrect: a is at offset one.

B - Incorrect: The PHP delimiter and string access are valid.

C - Incorrect: Interpolation emits the value of $var, not its name.

D - Correct: r occupies offset two of maroon.

Primary note proof

Study-note section WEB24

> String offsets start at zero: "maroon"[2] is r when the surrounding PHP is valid.

Study-note section WEB24

> Double-quoted strings interpolate variables and escapes such as \n; single quotes normally preserve dollar names literally.

Study-note section WEB23

> Use <?php ...

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 63 | lines 27-33

> 5.4 Data Types and Variables in PHP
> PHP is a very loosely typed language. This means that variables do not have to be declared before they are used, and that PHP always converts variables to the type required by their context when they are accessed. Variables can store data of different types, and different data types can do different things. PHP supports the following data types: String, Integer, Float (floating point numbers - also called double), Boolean, Array, Object, NULL and Resource

Web Programming module | PDF page 70 | lines 2-8

> We can also create arrays by adding the index of the item in between the square brackets. This will enable us to add items in unorderly fashion
> <?php
> $sample[0] = "sample 1";
> $sample[1] = "sample 2";

```
$sample[2] = "sample 3";
$sample[3] = "sample 4";
```

Question sources

Addis Ababa University mock exam · Question 63

Question 15 | Q00162

In PHP, which function initiates a session?

A. start()

B. s_start()

C. session_start()

D. session_str()

Hint

Follow the identifier and server-side state through the stated lifecycle or access check; distinguish client-controlled claims from server decisions.

Answer C: session_start()

The Starting a PHP Session example explicitly calls session_start() before storing information (p.76). Evidence: PDF p. 76.

Why each choice is right or wrong

A - Incorrect: Not the session initializer named by the module.

B - Incorrect: Not the documented name.

C - Correct: Exact documented initializer.

D - Incorrect: Not the documented name.

Primary note proof

Study-note section WEB29

session_start() starts/resumes PHP session handling and makes $_SESSION available.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 76 | lines 6-16

Starting a PHP Session
Before you can store user information in your PHP session, you must first start up the session.
<?php session_start(); ?>
<html>

<body>.....
</body>
</html>
The code above will register the user's session with the
server, allow you to start saving user information, and assign a UID for that user's
session.

Question sources

Addis Ababa University mock exam · Question 64

Question 16 | Q00163

Which single option gives the complete selection of the listed PHP superglobals?

A. $_GET

B. $_POST

C. $_SERVER

D. All of the above

Hint

Identify the data source or security responsibility precisely, and check whether a combined source or a different control could also qualify.

Answer D: All of the above

The names $_GET, $_POST and $_SERVER are all PHP superglobals, representing query values, supported form POST data and request/environment details respectively. The question asks for the complete selection in one option, so All of the above is the best response. Each individual array is genuinely a superglobal, but selecting just one would omit the other two valid entries.

Why each choice is right or wrong

A - Incorrect: True individually, but incomplete because POST and SERVER are also superglobals.

B - Incorrect: True individually, but incomplete because GET and SERVER are also superglobals.

C - Incorrect: True individually, but incomplete because GET and POST are also superglobals.

D - Correct: This aggregate includes all three valid superglobals.

Primary note proof

Study-note section WEB27

$_GET contains query parameters; $_POST contains supported form POST data; $_FILES upload metadata; $_SERVER request/environment details; $_COOKIE cookie values; $_SESSION session data after startup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 62 | lines 12-26

> PHP provides a large number of predefined variables to any script which it runs.
> PHP provides an additional set of predefined arrays containing variables from the web server the environment, and user input. These new arrays are called superglobals –
> All the following variables are automatically available in every scope.
> - $GLOBALS → Contains a reference to every variable which is currently available within the global scope of the script. The keys of this array are the names of the global variables.
> - $_SERVER → This is an array containing information such as headers, paths, and script locations. The entries in this array are created by the web server. There is no guarantee that every web server will provide any of these. See next section for a complete list of all the SERVER variables.
> - $_GET → An associative array of variables passed to the current script via the HTTP GET method.
> - $_POST → An associative array of variables passed to the current script via the HTTP POST method.

Question sources

Addis Ababa University mock exam · Question 65

Question 17 | Q00265

Which of the following is the correct syntax to link an external style sheet in the HTML file?

A. <link rel="stylesheet" src="style.css" />

B. <link rel="stylesheet" href="style.css" />

C. <style rel="stylesheet" src="style.css" />

D. <style rel="stylesheet" link="style.css" />

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer B: <link rel="stylesheet" href="style.css" />

The external CSS example uses <link rel="stylesheet" href="style.css"> (p.29). B matches the element, relationship and URL attribute. Evidence: PDF p. 29, PDF p. 28.

Why each choice is right or wrong

A - Incorrect: src is not the demonstrated stylesheet URL attribute.

B - Correct: Matches the complete link/rel/href pattern.

C - Incorrect: style is used for embedded rules, and src is wrong here.

D - Incorrect: style with link does not implement the module external-link syntax.

Primary note proof

Study-note section WEB11

> An external sheet uses <link rel="stylesheet" href="styles.css"> in head; an internal sheet uses style; an inline declaration uses an element’s style attribute.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 29 | lines 2-6

> E.g.
> <link rel="stylesheet" href="style.css">
> The rel attributes names the relationship of the linked document to the current document. The href attribute specifies the URL of the linked document. The URL can be relative or absolute

Web Programming module | PDF page 28 | lines 12-14

> 2. Embedded stylesheet
> This type of stylesheet should be used when a single document has a unique style. You define internal styles in the head section by using the <style> tag.

Question sources

Mock exam 4 · Question 67

Question 18 | Q00266

Which of the following is used for concatenation in PHP?

A. *(Asterisk)

B. . (dot)

C. append()

D. + (plus)

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: . (dot)

PHP examples concatenate labels, variables and HTML strings with dot, for example echo "name:".$name."
" (p.72). Evidence: PDF p. 72, PDF p. 64.

Why each choice is right or wrong

A - Incorrect: Asterisk is arithmetic, not the joining operator in the example.

B - Correct: Dot joins the strings and variables in the PHP example.

C - Incorrect: No append() string operation is established.

D - Incorrect: Plus performs numeric addition in the p.64 example, not the demonstrated string joining.

Primary note proof

Study-note section WEB24

> Dot concatenates; + is arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 72 | lines 18-22

```
Example:
function info($name,$age,$salary){
echo “name:”.$name.”<br>”;
echo “age:”.$age.”<br>”;
echo “salary:”.$salary.”<br>”; }
```

Web Programming module | PDF page 64 | lines 33-36

```
$first=6;
$second=5;
$result=$first+$second;
Echo $result;
```

Question sources

Mock exam 4 · Question 68

Question 19 | Q00268

Which one of the following is not used to generate dynamic web pages?

A. JSP

B. PHP

C. CSS

D. ASP.NET

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: CSS controls presentation rather than generating server-side pages.

WEB01 assigns CSS presentation and identifies PHP, JSP/Java, and ASP.NET applications as generating server responses. In this role distinction, CSS is the exclusion: it styles a representation rather than executing the server page-generation logic. CSS can affect displayed appearance, but that does not make it a server-side dynamic-page generator like the other offered technologies.

Why each choice is right or wrong

A - Incorrect: JSP/Java applications are identified in WEB01 as server-side response generators.

B - Incorrect: PHP applications can generate server responses and access databases, fitting dynamic-page generation.

C - Correct: CSS supplies presentation rules rather than the server logic that generates dynamic page representations.

D - Incorrect: ASP.NET applications are identified as server-side response generators, so they are not the exclusion.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 26 | lines 4-10

> In the previous chapter we have seen how to structure a web page using HTML.
> Now it is time to add presentation to our sites, to do that we will be using a technology
> called CSS. CSS (Cascading Style Sheets) is the code that styles web content. Using
> CSS you can apply styles to make web pages look exactly how you want. This works
> because CSS is connected to the DOM(Document Object Model). CSS is not a
> programming language. It's not a markup language either. CSS is a style sheet
> language. CSS is what you use to selectively style HTML elements.

Web Programming module | PDF page 60 | lines 4-10

> PHP is a server-side scripting language, which can be embedded in HTML or used as
> a standalone binary (although the former use is much more common). PHP stands for
> Hypertext Preprocessor. The product was originally named Personal Home Page Tools,
> and many people still think that is what the acronym stands for. But as it expanded in
> scope, a new and more appropriate name was selected by community vote. PHP is the
> language that you use to make the server generate dynamic output-output that is
> potentially different each time a browser re-quests a page.

Question sources

Mock exam 4 · Question 70

Question 20 | Q00314

In PHP MySQLi, after mysqli_init() and mysqli_ssl_set(...) have configured the required TLS and certificate-verification settings, which function initiates the database connection?

A. mysqli_real_connect(...)

B. pdo_connect_ssl()

C. mysql_ssl_connect()

D. secure_connect()

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer A: mysqli_real_connect(...)

MySQLi separates initialization and TLS configuration from opening the connection. Under the stated setup, mysqli_real_connect performs the connection step using the configured handle and required options. The other names are not standard connection functions in the supplied note. Choosing the actual API does not mean that any call automatically verifies certificates; that protection depends on the verification configuration explicitly included in the question.

Why each choice is right or wrong

A - Correct: This is the connection step in the documented MySQLi TLS sequence.

B - Incorrect: This is not the documented PDO or MySQLi connection API.

C - Incorrect: The note rejects this invented SSL connection function.

D - Incorrect: The note rejects this generic invented function name.

Primary note proof

Study-note section WEB31

> Configure MySQLi TLS options through mysqli_init(), mysqli_ssl_set(...) and mysqli_real_connect(...) with the required verification configuration.

Study-note section WEB31

> mysqli_ssl_connect(), pdo_connect_ssl(), mysql_ssl_connect() and secure_connect() are not substitutes for the MySQLi TLS configuration sequence above.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 79 | lines 9-16

5.13.2 Creating Database Connection in PHP
1. Connecting to MySQL
The basic command to initiate a MySQL connection is
$link=mysql_connect($hostname, $user, $password); if you‘re using variables, or
$link=mysql_connect(„localhost“, „root“, „sesame“); if you‘re using literal strings. Here
$Hostname - the ip address or hostname of the MySQL server. If the MySQL server
and the php server are on the same machine use localhost or 127.0.0.1, else use the ip
address of the database server.

Question sources

Computer Science model exam · Question 45

University of Gondar model exam · Question 45

Question 21 | Q00315

Which JavaScript function is used to display a dialog box with an alert message?

A. console.log()

B. confirm()

C. alert()

D. prompt()

Hint

Distinguish the browser object or dialog role from the kind of response it collects.

Answer C: alert()

Alert shows a browser popup containing its argument (p.47). Prompt requests input, confirm returns a Boolean choice, and console.log writes to the console (p.46). Evidence: PDF p. 47, PDF p. 46.

Why each choice is right or wrong

A - Incorrect: Console output is not an alert dialog.

B - Incorrect: Confirmation presents OK/Cancel and returns a Boolean.

C - Correct: Exactly the documented alert popup.

D - Incorrect: Prompt additionally requests text input.

Primary note proof

Study-note section WEB22

alert displays information, confirm returns a Boolean, and prompt requests text or returns null on cancellation.

Study-note section WEB22

console.log logs diagnostics.

Study-note section WEB-S03

Alert is a notification; confirm additionally asks for an OK/Cancel decision, while prompt additionally offers a text field.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 47 | lines 2-9

2. Alert: shows a browser pop up with the information inside the parenthesis
3. The write() method
- The write() method writes HTML expressions or JavaScript code to a document.
- The write() method is mostly used for testing: If it is used after an HTML document is fully loaded, it will delete all existing HTML.
Usage:
document.write(expr)

Web Programming module | PDF page 46 | lines 16-27

Accepting user input in JavaScript actually is not a difficult task. JavaScript has a function called prompt, which is used to accept input from the user. But for the result to be useful for us, it has to be stored in a variable. Here is a simple example
var name = prompt("Enter your name: ");
alert("Hello " + name);
➢ Prompt will show a modal windows with a text message, an input field for the visitor, and the buttons OK/Cancel
➢ We can also use confirm to get information from a user:- the function confirm shows a modal window with a question and two buttons: OK and Cancel
The result is true if OK is pressed and false otherwise
Generating an output in JavaScript can be done in different ways.
1. Console.log: outputs a string to the browser console.

Question sources

Computer Science model exam · Question 46

University of Gondar model exam · Question 46

Question 22 | Q00316

Which JavaScript method is used to select an HTML element by its ID?

A. getElementById()

B. getElementByTag()

C. getElementByName()

D. getElementByClass()

Hint

Track which node is selected or created, and distinguish changing content from attaching a node.

Answer A: getElementById()

The DOM methods list says getElementById(id) returns the element with that specific ID (p.52). Evidence: PDF p. 52.

Why each choice is right or wrong

A - Correct: Exact ID lookup method.

B - Incorrect: Not the documented getElementsByTagName; also tag selection differs.

C - Incorrect: No getElementByName is defined here; not ID lookup.

D - Incorrect: Not the documented class lookup and not ID lookup.

Primary note proof

Study-note section WEB20

> document.getElementById("x") returns the matching element or null; querySelector(".item") returns the first match, and querySelectorAll returns a static NodeList.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 52 | lines 10-19

> 4.10.1 Accessing HTML elements in JavaScript
> In the DOM Api, there are different methods to get access to HTML elements. We will list the available methods here, but only the most recent methods are covered here.
> - getElementById(id) returns an element with that specific ID
> - getElmentsByClassName(class) returns elements with that specific class
> getElementsByTagName(tag) returns elements with the given tag name
> - querySelector(selector) → The Document method querySelector() returns the first Element within the document that matches the specified selector, or group of selectors. If no matches are found, null is returned.

Question sources

Computer Science model exam · Question 47

University of Gondar model exam · Question 47

Question 23 | Q00317

Which language is used to add interactivity and dynamic behavior to a web page?

A. HTML

B. PHP

C. CSS

D. JavaScript

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: JavaScript

The module divides roles: HTML structure, CSS presentation, JavaScript functionality and interactive content updates (p.44). Evidence: PDF p. 44, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: HTML supplies structure.

B - Incorrect: PHP generates server output, whereas this stem targets page interactivity.

C - Incorrect: CSS supplies presentation.

D - Correct: JavaScript supplies interactive browser behavior.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 9-19

> JavaScript is a scripting or programming language that allows us to implement complex features on web pages. JavaScript will enable us to display timely content updates, interactive maps, 2D/3D animated graphics etc. JavaScript is one of the three layers of web technologies which lets to add functionality to a web page. The three layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'
> What can JavaScript really do? It can

- Stores values inside variables
- Perform operations on pieces of texts
- Run a code in response to events

Web Programming module | PDF page 60 | lines 4-10

PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science model exam · Question 48

University of Gondar model exam · Question 48

Question 24 | Q00318

Which of the following languages is primarily responsible for defining the structure and content of a web page?

A. HTML

B. JavaScript

C. PHP

D. CSS

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: HTML

HTML defines the meaning and structure of web content; CSS handles presentation and JavaScript behavior (p.10). Evidence: PDF p. 10, PDF p. 60.

Why each choice is right or wrong

A - Correct: Precisely the module HTML role.

B - Incorrect: Behavior rather than primary structure.

C - Incorrect: Server-generated dynamic output, not the markup language defining structure.

D - Incorrect: Presentation rather than primary content structure.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 10 | lines 9-16

> HTML (HyperText Markup Language) is the most basic building block of the Web.
> HTML is not a programming language rather it is a markup language that tells web browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
> HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Web Programming module | PDF page 60 | lines 4-10

> PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science model exam · Question 49

University of Gondar model exam · Question 49

Question 25 | Q00319

Which PHP function is used to check if a file or directory exists?

A. file_exists()

B. file_exists_check()

C. is_file()

D. file_check()

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: file_exists()

WEB30 directly distinguishes file_exists, which tests files or directories, from is_file, which narrows the test to regular files. The question explicitly includes either a file or directory, so file_exists matches. The other two named checks are not the provided APIs, while is_file is too restrictive for the requested test.

Why each choice is right or wrong

A - Correct: file_exists() checks existence for files or directories, exactly the two entry kinds in the stem.

B - Incorrect: file_exists_check() is not the filesystem existence API identified by the complete note.

C - Incorrect: is_file() tests regular files and would not report an existing directory as a regular file.

D - Incorrect: file_check() is not the filesystem existence API given in WEB30.

Primary note proof

Study-note section WEB30

> file_exists tests files or directories; is_file narrows to regular files.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 81 | lines 22-27

> 5.14 Input-Output
> PHP has several functions for creating, reading, uploading, and editing files. The
> fopen() function is used to open files in PHP.
> Opening a file
> The first parameter of this function contains the name of the file to be opened and
> the second parameter specifies in which mode the file should be opened: Example

Question sources

Computer Science model exam · Question 50

University of Gondar model exam · Question 50

Question 26 | Q00320

Using the procedural MySQLi API with connection $conn, which function reports rows changed, inserted or deleted by the last successful data-changing statement?

A. mysqli_num_rows($result)

B. $stmt->rowCount()

C. get_affected_rows($conn)

D. mysqli_affected_rows($conn)

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer D: mysqli_affected_rows($conn)

The procedural MySQLi affected-row function takes a connection and reports the effect of a successful data-changing statement under the connection's options. Counting rows in a SELECT result is a different operation, and a PDO statement method belongs to a different API. The question specifies both the API and the kind of row count, making the procedural call uniquely appropriate without assuming all PHP database interfaces share identical function names.

Why each choice is right or wrong

A - Incorrect: This counts rows in a result set, not the last data-changing statement's effect.

B - Incorrect: This is a PDO statement method rather than the requested procedural MySQLi call.

C - Incorrect: This is not the documented procedural function.

D - Correct: This is the documented affected-row function for the connection.

Primary note proof

Study-note section WEB32

> mysqli_num_rows counts a result's rows; mysqli_affected_rows($conn) counts changed/inserted/deleted rows under its options; mysqli_insert_id($conn) reports a generated AUTO_INCREMENT value from the connection.

Study-note section WEB-S04

> PDOStatement::rowCount() is a method on a PDO statement object that reports affected rows for its DELETE, INSERT or UPDATE statement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 80 | lines 16-23

> 5.13.4 Processing Query Result.
> 4. Fetching data sets
> The result of a query returned by the mysql_query() function is an interger showing the

success or failure of the function not the desired data. So you have to use one of the mysql_fetch_..() functions to get the actual result. There are many functions to do so, but the most commonly used ones are- Mysql_fetch_row($result) - this function takes the result of the mysql_query() function as an argument and returns the data as an enumerated array.

Question sources

Computer Science model exam · Question 51

University of Gondar model exam · Question 51

Question 27 | Q00321

Which PHP superglobal array is used to retrieve information about the server and execution environment?

A. $_POST

B. $_SESSION

C. $_GET

D. $_SERVER

Hint

Identify the data source or security responsibility precisely, and check whether a combined source or a different control could also qualify.

Answer D: $_SERVER

$_SERVER is the predefined array for server-created headers, paths and script locations (p.62). Evidence: PDF p. 62, PDF p. 76.

Why each choice is right or wrong

A - Incorrect: POST contains form/request variables.

B - Incorrect: SESSION stores per-user application state (p.76).

C - Incorrect: GET contains GET request variables.

D - Correct: Matches server/execution information.

Primary note proof

Study-note section WEB27

$_GET contains query parameters; $_POST contains supported form POST data; $_FILES upload metadata; $_SERVER request/environment details; $_COOKIE cookie values; $_SESSION session data after startup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 62 | lines 19-26

- $_SERVER → This is an array containing information such as headers, paths, and script locations. The entries in this array are created by the web server. There is no guarantee that every web server will provide any of these. See next section for a complete list of all the SERVER variables.
- $_GET → An associative array of variables passed to the current script via the HTTP GET method.
- $_POST → An associative array of variables passed to the current script via the HTTP POST method.

Web Programming module | PDF page 76 | lines 17-23

Storing a Session Variable
The correct way to store and retrieve session variables is to use the PHP $_SESSION variable:

```
<?php
session_start();
// store session data
$_SESSION['views']=1;
```

Question sources

Computer Science model exam · Question 52

University of Gondar model exam · Question 52

Question 28 | Q00322

Which PHP superglobal is the dedicated source for supported form fields submitted using HTTP POST, rather than a configurable combination of request sources?

A. $_REQUEST

B. $_GET

C. $_POST

D. $_SERVER

Hint

Identify the data source or security responsibility precisely, and check whether a combined source or a different control could also qualify.

Answer C: $_POST

$_POST is the dedicated array for supported form POST data. $_REQUEST may also contain POST values, but combines sources according to configuration and is therefore not the dedicated source requested. $_GET contains query parameters, while $_SERVER contains request and environment details. The distinction is the origin-specific array, not whether a value could appear in more than one superglobal.

Why each choice is right or wrong

A - Incorrect: $_REQUEST can include POST values, but combines configured request sources rather than identifying only POST form data.

B - Incorrect: $_GET contains URL query parameters rather than the dedicated POST form map.

C - Correct: $_POST is specifically the supported form POST-data source.

D - Incorrect: $_SERVER holds request/environment information, not the submitted POST field map.

Primary note proof

Study-note section WEB27

> $_GET contains query parameters; $_POST contains supported form POST data; $_FILES upload metadata; $_SERVER request/environment details; $_COOKIE cookie values; $_SESSION session data after startup.

Study-note section WEB27

> $_REQUEST may combine GET/POST/cookie data according to configuration; it is not a database interface and has ambiguous precedence, so prefer the intended source.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 62 | lines 19-26

> - $_SERVER → This is an array containing information such as headers, paths, and script locations. The entries in this array are created by the web server. There is no guarantee that every web server will provide any of these. See next section for a complete list of all the SERVER variables.
> - $_GET → An associative array of variables passed to the current script via the HTTP GET method.
> - $_POST → An associative array of variables passed to the current script via the HTTP POST method.

Web Programming module | PDF page 73 | lines 11-18

> • $_GET[]: It is used the retrieve the information from the form control through the parameters sent in the URL. It takes the attribute given in the url as the parameter.
> • $_POST[]: It is used the retrieve the information from the form control through the HTTP POST method. IT takes name attribute of corresponding form control as the parameter.
> • $_REQUEST[]: It is used to retrieve an information while using a database.

Question sources

Computer Science model exam · Question 53

University of Gondar model exam · Question 53

Question 29 | Q00357

Which names describe a server that receives webpage requests through HTTP/HTTPS and returns representations? Select all that apply.

A. Http Server

B. Web Server

C. FTP server

D. DNS Server

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A, B: Http Server; Web Server

A web server is also called an HTTP server in this context. It receives requests and returns resource representations, possibly invoking application code to construct them. The two labels describe the same serving role and must both be selected. FTP serves a distinct transfer purpose, while DNS supplies name-to-record resolution. The question now explicitly permits the synonymous correct names instead of forcing an arbitrary choice between them.

Why each choice is right or wrong

A - Correct: HTTP server names the request-and-response role described.

B - Correct: Web server is the equivalent name in this context.

C - Incorrect: FTP is not the stated HTTP webpage-serving role.

D - Incorrect: DNS resolves names rather than returning webpage representations through HTTP.

Primary note proof

Study-note section WEB01

> A web/HTTP server receives requests and returns representations, sometimes invoking application code.

Study-note section WEB02

> Hosting supplies an environment where a site is served; domain registration reserves a name, and DNS maps names to network records.

Study-note section WEB-S14

> HTTP defines web application requests, response status codes and representations.

Study-note section WEB-S14

> An HTTP exchange can use underlying TCP connections and IP addresses; naming that network stack does not identify the application-level method or status semantics.

Study-note section WEB-S14

FTP is a file-transfer application protocol.

Study-note section WEB-S14

SMTP transports electronic mail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 9 | lines 2-4

• Web pages are files stored on computers called Web servers.
• Computers reading the Web pages are called Web clients.
• Web clients view the pages with a program called a Web browser.

Web Programming module | PDF page 8 | lines 29-34

• The World Wide Web (WWW) is most often called the Web.
• The Web is a network of computers all over the world.
• All the computers in the Web can communicate with each other.
• All the computers use a communication standard called HTTP.
How does the WWW work?
• Web information is stored in documents called Web pages.

Question sources

Computer Science model exam · Question 88

University of Gondar model exam · Question 87

Question 30 | Q00722

The attribute used in the body tag to change the default font color for the page is _____.

A. Bgcolor attribute

B. text attribute

C. color attribute

D. link tag

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer B: text attribute

The body attribute list explicitly gives text="text color" separately from bgcolor and link (p.12). Evidence: PDF p. 12.

Why each choice is right or wrong

A - Incorrect: bgcolor sets background color.

B - Correct: text is explicitly paired with text color.

C - Incorrect: color is not the listed body default-text attribute.

D - Incorrect: link is paired with link color, not all text.

Primary note proof

Study-note section WEB13

> In historical markup, the body text attribute controlled default text color, while bgcolor controlled background.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 12 | lines 30-36

> Some of the common attributes used in the body element
> ✓ bgcolor=“color”
> ✓ background=“img url”
> ✓ text=“text color”
> ✓ link=“link color”
> ✓ alink=“active link color”
> ✓ vlink=“visited link color”.....

Question sources

Computer Science model exam 2 · Question 63

Question 31 | Q00723

For a basic static website with no client scripts that change its displayed content, which single option gives the complete set of true statements about its stored-page delivery model?

A. Changing the stored page representation requires changing the stored file

B. The server returns stored files rather than constructing a response from application logic

C. The page representation is delivered as stored

D. Viewers requesting the same unchanged file receive the same stored representation

E. All of the above.

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer E: All of the above.

A basic static server returns stored files rather than dynamically constructing representations from application logic. The question restricts content to the stored representation and excludes scripts that could change the rendered view. Under those assumptions, A through D describe compatible aspects of the same model. All of the above is the complete single selection; each individual statement is true but would cover only part of the requested set.

Why each choice is right or wrong

A - Incorrect: True but incomplete: it states only how the stored representation changes.

B - Incorrect: True but incomplete: it states the serving model alone.

C - Incorrect: True but incomplete: it omits the other delivery properties.

D - Incorrect: True but incomplete: it states sameness for the same unchanged resource only.

E - Correct: This aggregate contains every true statement under the specified model.

Primary note proof

Study-note section WEB02

> A basic static server returns stored files; dynamic processing constructs a representation from logic, requests or data.

Study-note section WEB02

> Static pages can still run interactive browser scripts, and dynamic generation need not change visible text every time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 8 | lines 3-6

1.1 Course overview
This chapter deals with the Overview of Internet, website, webpage, types of websites (Static vs dynamic), webpage design tools, Web World Wide Web, Web Hosting and Domain Name Registration. To deliver these contents active learning methods such as

Web Programming module | PDF page 60 | lines 8-10

scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science model exam 2 · Question 64

Question 32 | Q00724

Which single option gives the complete selection of technologies that can generate server-side web responses?

A. PHP

B. ASP.NET

C. Python

D. All of the above

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: All of the above

PHP, Python and ASP.NET applications can generate server responses and access databases. The question deliberately asks about technologies rather than calling every entry a programming language, since ASP.NET is an application technology. All three listed entries satisfy that broader but precise role, so the aggregate is the complete single answer. Selecting an individual entry would state a true example while omitting the other two supported server-side technologies.

Why each choice is right or wrong

A - Incorrect: PHP fits, but this single entry omits the other valid technologies.

B - Incorrect: ASP.NET applications fit, but the selection is incomplete.

C - Incorrect: Python applications fit, but the selection is incomplete.

D - Correct: All three entries support server response generation.

Primary note proof

Study-note section WEB01

PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 3-10

5.1 Introduction to PHP
PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science model exam 2 · Question 65

Question 33 | Q00725

Which option gives the complete set of listed JavaScript dialogs that can display a supplied message, whether or not they also collect a response?

A. Prompt

B. Alert

C. Confirm

D. All of the above

Hint

Distinguish the browser object or dialog role from the kind of response it collects.

Answer D: All of the above

Alert, confirm and prompt all present a supplied message. Alert provides a notification, confirm additionally requests an OK/Cancel decision, and prompt additionally offers a text field. The question asks for the complete set of qualifying dialogs, so All of the above is the complete selection. Each individual dialog is a true example but omits the other two qualifying dialogs.

Why each choice is right or wrong

A - Incorrect: Prompt displays a message while requesting text, so it qualifies; selecting it alone omits alert and confirm and is incomplete.

B - Incorrect: Alert displays a notification, so it qualifies; selecting it alone omits prompt and confirm and is incomplete.

C - Incorrect: Confirm displays a message while requesting an OK/Cancel decision, so it qualifies; selecting it alone omits prompt and alert and is incomplete.

D - Correct: All three listed dialogs can display a supplied message, so this is the complete qualifying set.

Primary note proof

Study-note section WEB-S03

> alert(message), confirm(message) and prompt(message) all present a supplied message.

Study-note section WEB22

> alert displays information, confirm returns a Boolean, and prompt requests text or returns null on cancellation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 47 | lines 2-5

> 2. Alert: shows a browser pop up with the information inside the parenthesis
> 3. The write() method
> - The write() method writes HTML expressions or JavaScript code to a document.

Web Programming module | PDF page 46 | lines 16-25

> Accepting user input in JavaScript actually is not a difficult task. JavaScript has a function called prompt, which is used to accept input from the user. But for the result to be useful for us, it has to be stored in a variable. Here is a simple example
> var name = prompt("Enter your name: ");
> alert("Hello " + name);
> ➢ Prompt will show a modal windows with a text message, an input field for the visitor, and the buttons OK/Cancel
> ➢ We can also use confirm to get information from a user:- the function confirm shows a modal window with a question and two buttons: OK and Cancel
> The result is true if OK is pressed and false otherwise

Question sources

Computer Science model exam 2 · Question 66

Question 34 | Q00726

What does these code possibly refers to<link rel="stylesheet" href="styles.css"> ?

A. Defines a new CSS by the name styles

B. Calls an internal style sheet called styles

C. Calls an external style sheet called styles

D. Defines an inline CSS by the name styles

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer C: Calls an external style sheet called styles

The link tag identifies a separate stylesheet URL (p.29); p.28 explains that this links an external stylesheet shared by pages. Evidence: PDF p. 29, PDF p. 28.

Why each choice is right or wrong

A - Incorrect: It references an existing file, not a CSS definition.

B - Incorrect: Internal CSS instead uses a style block.

C - Correct: It links the external styles.css file.

D - Incorrect: Inline CSS instead uses an element style attribute.

Primary note proof

Study-note section WEB11

> An external sheet uses <link rel="stylesheet" href="styles.css"> in head; an internal sheet uses style; an inline declaration uses an element’s style attribute.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 29 | lines 2-6

> E.g.
> <link rel="stylesheet" href="style.css">
> The rel attributes names the relationship of the linked document to the current document. The href attribute specifies the URL of the linked document. The URL can be relative or absolute

Web Programming module | PDF page 28 | lines 27-31

> 3. External style sheets
> An external style sheet is used when the style is applied to many pages. With an external style sheet, you can change the look of an entire website by changing one file (the style sheet). Each page should be linked to the style sheet using the <link> tag inside the head section.

Question sources

Computer Science model exam 2 · Question 67

Question 35 | Q00727

Which valid HTML snippet uses CSS to set the body background color to yellow?

A. <body background="yellow">

B. <background>yellow</background>

C. <body style="background-color: yellow">

D. <background color="yellow">text<background>

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer C: <body style="background-color: yellow">

The CSS property background-color controls an element's background. An inline declaration belongs in the opening element's style attribute, so the body tag applies the requested color. The other snippets either use a background attribute instead of the specified CSS property or invent markup that does not express this body style.

Why each choice is right or wrong

A - Incorrect: This uses an attribute rather than the requested background-color declaration.

B - Incorrect: This invents a background element instead of styling body.

C - Correct: The valid body start tag contains the appropriate CSS declaration.

D - Incorrect: This markup does not supply a valid body style declaration.

Primary note proof

Study-note section WEB13

> color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Study-note section WEB11

> An external sheet uses <link rel="stylesheet" href="styles.css"> in head; an internal sheet uses style; an inline declaration uses an element's style attribute.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 28 | lines 4-8

> 1. Inline stylesheet
> In HTML, style information can be specified for an individual element via the style attribute. The value of a style attribute is a declaration block without the curly braces.
> E.g.
> Inline style

Web Programming module | PDF page 34 | lines 3-10

3.3.1 Foreground and Background Properties
The background shorthand CSS property sets all background style properties at once, such as color, image, origin and size, or repeat method. You have also specific background properties that you can apply like background-color, background-image etc.
To set the background-color of an element and provide the value of the color in different form.
1. Keyword values -> red, blue

Web Programming module | PDF page 12 | lines 30-33

Some of the common attributes used in the body element
✓ bgcolor="color"
✓ background="img url"
✓ text="text color"

Question sources

Computer Science model exam 2 · Question 68

Question 36 | Q00728

What is the method used to get the result from the form data sent with both GET and POST?

A. $_REQUEST

B. $_POST

C. $_GET

D. A & B

Hint

Identify the data source or security responsibility precisely, and check whether a combined source or a different control could also qualify.

Answer A: $_REQUEST can combine GET and POST input according to configuration.

WEB27 states that $_GET contains query parameters and $_POST supported form POST data, while $_REQUEST may combine GET, POST, and cookie data according to configuration. Therefore $_REQUEST is the offered combined-source mechanism. It is an array rather than a method, and its configuration-dependent precedence is a reason to prefer an explicit source in real code.

Why each choice is right or wrong

A - Correct: $_REQUEST can combine GET and POST values according to PHP configuration, matching the combined-source question.

B - Incorrect: $_POST holds supported form POST data and does not generally combine query GET values.

C - Incorrect: $_GET holds URL query parameters rather than combining both query and form POST sources.

D - Incorrect: Adding $_POST to the already combined $_REQUEST is redundant and is not the single combined mechanism requested.

Primary note proof

Study-note section WEB27

> $_GET contains query parameters; $_POST contains supported form POST data; $_FILES upload metadata; $_SERVER request/environment details; $_COOKIE cookie values; $_SESSION session data after startup.

Study-note section WEB27

> $_REQUEST may combine GET/POST/cookie data according to configuration; it is not a database interface and has ambiguous precedence, so prefer the intended source.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 73 | lines 11-18

> • $_GET[]: It is used the retrieve the information from the form control through the parameters sent in the URL. It takes the attribute given in the url as the parameter.
> • $_POST[]: It is used the retrieve the information from the form control through the HTTP POST method. IT takes name attribute of corresponding form control as the parameter.
> • $_REQUEST[]: It is used to retrieve an information while using a database.

Question sources

Computer Science model exam 2 · Question 69

Question 37 | Q00729

When using mysqli_connect(), which argument selects the default database and may be deferred until after connecting?

A. Host

B. Database

C. Username

D. Password

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer B: Database

The mysqli_connect signature separates host, user, password and database, followed by port and socket. Selecting a database can be deferred even though several arguments also have defaults. The revised question asks which argument performs that particular selection, rather than incorrectly implying only one argument is optional. The database argument names the initial database; the remaining listed arguments identify the server or provide authentication credentials.

Why each choice is right or wrong

A - Incorrect: The host identifies the server, not the default database.

B - Correct: This selects the initial database and can be deferred.

C - Incorrect: The username supplies an authentication identity.

D - Incorrect: The password supplies a credential, not the database selection.

Primary note proof

Study-note section WEB31

> mysqli_connect(host,user,password,database,port,socket) opens a connection; selecting a database can be deferred, and defaults make several arguments technically optional.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 79 | lines 9-29

> 5.13.2 Creating Database Connection in PHP
> 1. Connecting to MySQL
> The basic command to initiate a MySQL connection is
> $link=mysql_connect($hostname, $user, $password); if you‘re using variables, or
> $link=mysql_connect(„localhost“, „root“, „sesame“); if you‘re using literal strings. Here
> $Hostname - the ip address or hostname of the MySQL server. If the MySQL server and the php server are on the same machine use localhost or 127.0.0.1, else use the ip address of the database server.
> $User - the user name of the database if specified by one of the grant commands. If username and password is not given to the database (which is not recommended) use root as your username which is the default.
> $Password - the password of the database if specified by one of the grant commands. The default password for any MySQL database is empty, so if password is not granted leave it null. $link - is a variable and is called the link identifier. You will use this variable later.
> 2. Selecting the database
> After creating the connection identifier, the next thing, you will do is select a database among the databases you have created. To do this you will use the mysql_select_db() method.
> Syntax:
> $db=mysql_select_db(„dbname“,[link identifier]);

Question sources

Computer Science model exam 2 · Question 70

Question 38 | Q00888

Consider the CSS reset:
nav ul {
list-style-type: none;
margin: 0;
padding: 0;
}
What does this rule itself accomplish for a navigation list?

A. Aligns the navigation bar to the right

B. Adds hover effects to list items

C. Removes list bullets and default margin and padding

D. Converts the list into a dropdown menu

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer C: Removes list bullets and default margin and padding

The rule selects unordered lists inside nav elements. list-style-type removes list markers, while margin and padding reset the relevant spacing to zero. Nothing in the rule chooses a horizontal or right-aligned layout, installs a hover state or implements dropdown behavior.

Why each choice is right or wrong

A - Incorrect: No alignment property is supplied.

B - Incorrect: No hover selector or interaction rule is supplied.

C - Correct: These are exactly the marker and spacing effects of the three declarations.

D - Incorrect: A dropdown needs additional layout and interaction rules.

Primary note proof

Study-note section WEB08

> Resetting nav ul with list-style-type:none; margin:0; padding:0 removes bullets and default spacing only; it does not itself arrange links horizontally or create a dropdown.

Study-note section WEB11

> Type p selects paragraph elements, .notice selects a class, #logo an ID, h1,h2 groups selectors, nav ul selects descendants and nav>ul direct children.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 39 | lines 2-10

tag. Lists by default have a list-style-type, margin and padding properties set by the user agent style sheet. Since we don't want the default styling lets reset it
Here is the code to do that
ul {
list-style-type: none;
margin: 0;
padding: 0;
}
Next set the display property of ul to flex. This will align the list items

Question sources

Computer Science Exit Exam · 2018 · Question 1

Question 39 | Q00919

Which PHP superglobal is dedicated specifically to retrieving URL query parameters, rather than a configurable mixture of request sources?

A. $_REQUEST

B. $_POST

C. $_GET

D. isset()

Hint

Identify the data source or security responsibility precisely, and check whether a combined source or a different control could also qualify.

Answer C: $_GET

URL query values belong to $_GET. $_REQUEST can also include query data under some configurations, but it combines sources and therefore is not the dedicated query array requested here. $_POST represents supported form POST data, and isset tests whether an entry exists and is non-null rather than retrieving the entry's value. The question asks for the dedicated query array, making $_GET the unique selection.

Why each choice is right or wrong

A - Incorrect: This can mix query, POST and cookie sources according to configuration.

B - Incorrect: This represents form POST data rather than the dedicated URL query source.

C - Correct: This is the dedicated query-parameter superglobal.

D - Incorrect: This tests existence and non-null; it is not a request-data array.

Primary note proof

Study-note section WEB27

$_GET contains query parameters; $_POST contains supported form POST data; $_FILES upload metadata; $_SERVER request/environment details; $_COOKIE cookie values; $_SESSION session data after startup.

Study-note section WEB27

$_REQUEST may combine GET/POST/cookie data according to configuration; it is not a database interface and has ambiguous precedence, so prefer the intended source.

Study-note section WEB27

isset tests existence and non-null, not truth or valid syntax.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 73 | lines 9-18

• isset(): This function is used to determine whether the variable or a form control is having a value or not.
• $_GET[]: It is used the retrieve the information from the form control through the parameters sent in the URL. It takes the attribute given in the url as the parameter.
• $_POST[]: It is used the retrieve the information from the form control through the HTTP POST method. IT takes name attribute of corresponding form control as the parameter.
• $_REQUEST[]: It is used to retrieve an information while using a database.

Question sources

Computer Science Exit Exam · 2018 · Question 32

Question 40 | Q00926

In CSS, which property is used to change the text color of an element?

A. text-color

B. font-color

C. color

D. Foreground

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer C: color

The color property sets foreground content color (p.34), and p.31 illustrates p { color: red; }. Evidence: PDF p. 34, PDF p. 31.

Why each choice is right or wrong

A - Incorrect: Not the documented CSS name.

B - Incorrect: Not the documented CSS name.

C - Correct: Exact property used for text/foreground color.

D - Incorrect: A description of the effect, not the CSS property spelling.

Primary note proof

Study-note section WEB13

> color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 34 | lines 18-20

> Syntax:
> background-image: url(‘anyname.jpg’)
> The color property sets the color of the foreground content of the selected elements

Web Programming module | PDF page 31 | lines 2-4

```
p {
color: red;
}
```

Question sources

Computer Science Exit Exam · 2018 · Question 39

Question 41 | Q00946

is a service that allows organizations and individuals to post a website or a

web page onto the Internet.

A. Web hosting

B. Internet service provider

C. .

Web client

D. Domain name registration

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Web hosting

The module defines web hosting as the service allowing organizations/individuals to post a website or web page onto the Internet (p.9). Evidence: PDF p. 9.

Why each choice is right or wrong

A - Correct: Exact definition.

B - Incorrect: An ISP access service is not the hosting definition here.

C - Incorrect: A web client reads pages rather than hosting them.

D - Incorrect: Registration reserves a name, not the hosting service.

Primary note proof

Study-note section WEB02

> Hosting supplies an environment where a site is served; domain registration reserves a name, and DNS maps names to network records.

Study-note section WEB02

> An ISP provides connectivity.

Study-note section WEB01

> A browser/user agent requests resources and renders HTML/CSS, then runs permitted JavaScript.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 9 | lines 12-24

> What is web hosting?
> Web hosting is the place where all the files of your website live. Web hosting is a service that allows organizations and individuals to post a website or a web page onto the Internet. A web host, or web hosting service provider, is a business that provides the technologies and services needed for the website or webpage to be viewed in the Internet. Websites are hosted, or stored, on special computers called servers. When Internet users want to view your website, all they need to do is type your website address or domain (the domain name for the site has to be registered) into their browser. Their computer will then connect to your server and your webpages will be delivered to them through the browser.
> Domain Name Registration
> Domain name registration is the act of reserving a name on the Internet for a certain period. To get the address of a new website you have to register the domain name with

Question sources

Computer Science Exit Exam · 2018 · Question 59

Question 42 | Q00958

Which protocol is mostly used for communication between client and server in a web application?

A. FTP

B. SMTP

C. .
HTTP

D. SSH

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: .
HTTP

The WWW discussion explicitly says web computers use the HTTP communication standard (p.8). Evidence: PDF p. 8.

Why each choice is right or wrong

A - Incorrect: Not the module web communication standard.

B - Incorrect: Not the module web communication standard.

C - Correct: HTTP is expressly named.

D - Incorrect: Not the module web communication standard.

Primary note proof

Study-note section WEB01

> The Internet connects networks; the Web is a resource system commonly using HTTP.

Study-note section WEB-S14

> HTTP defines web application requests, response status codes and representations.

Study-note section WEB-S14

> An HTTP exchange can use underlying TCP connections and IP addresses; naming that network stack does not identify the application-level method or status semantics.

Study-note section WEB-S14

> FTP is a file-transfer application protocol.

Study-note section WEB-S14

> SMTP transports electronic mail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 8 | lines 29-34

• The World Wide Web (WWW) is most often called the Web.
• The Web is a network of computers all over the world.
• All the computers in the Web can communicate with each other.
• All the computers use a communication standard called HTTP.
How does the WWW work?
• Web information is stored in documents called Web pages.

Question sources

Computer Science Exit Exam · 2018 · Question 71

Question 43 | Q00960

What is the extension of JavaScript file?

A. javaS

B. JS

C. ¢.
java

D. CSS

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer B: JS

The external script example uses script.js, establishing the .js extension (p.45). B represents JS, ignoring case of the descriptive option. Evidence: PDF p. 45, PDF p. 44.

Why each choice is right or wrong

A - Incorrect: javaS is not the demonstrated extension.

B - Correct: JS corresponds to the demonstrated .js extension.

C - Incorrect: java is not the demonstrated extension; Java is a different language.

D - Incorrect: CSS is for stylesheets, not JavaScript.

Primary note proof

Study-note section WEB15

Use <script src="app.js"></script> for an external .js file; script content has no HTML closing-tag syntax inside its JavaScript source.

Study-note section WEB15

JavaScript originated at Netscape; it is distinct from Java and supports sophisticated applications through prototype-based objects and first-class functions.

Study-note section WEB11

An external sheet uses <link rel="stylesheet" href="styles.css"> in head; an internal sheet uses style; an inline declaration uses an element's style attribute.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 45 | lines 12-14

- We can also include JavaScript as an external file. We can use a similar syntax
like the internal and add the src attribute. Here is how
<script src="script.js" defer></script>

Web Programming module | PDF page 44 | lines 4-7

JavaScript (JS) is a lightweight, interpreted or compiled programming language. It is a single threaded, dynamic language, supporting object oriented, functional programming styles. Do not confuse JavaScript with the Java Programming language. The two-programming language have very different syntax, semantics, and use.

Question sources

Computer Science Exit Exam · 2018 · Question 73

Question 44 | Q00980

A developer stores $_SESSION["user_id"] server-side and an editable logged_in cookie client-side. A protected endpoint grants access solely when logged_in=1 and never validates the server-side session. An unauthenticated attacker sets that cookie. Which statement is most accurate?

A. The endpoint is vulnerable: trusting the cookie alone bypasses the required server-session authorization

B. The endpoint is secure because server-side session data exists

C. The endpoint is vulnerable only if the genuine session ID is exposed

D. The endpoint is secure if HTTPS is used

Hint

Follow the identifier and server-side state through the stated lifecycle or access check; distinguish client-controlled claims from server decisions.

Answer A: The endpoint is vulnerable: trusting the cookie alone bypasses the required server-session authorization

The protected endpoint trusts an attacker-controlled value rather than checking authenticated server-side state. Setting the flag is therefore sufficient to gain the access that should require session authorization. Merely storing a user ID elsewhere does not help when that state is never consulted. No genuine session identifier must be stolen for this bypass, and HTTPS protects transmission rather than proving that a client-generated authorization claim is true.

Why each choice is right or wrong

A - Correct: The stated cookie-only authorization path allows the attacker to bypass the required session check.

B - Incorrect: Unused server-side state does not protect this endpoint.

C - Incorrect: No genuine session ID is needed when the endpoint trusts only the flag.

D - Incorrect: Transport protection does not make a forged client flag trustworthy.

Primary note proof

Study-note section WEB29

> A client cookie logged_in=1 is attacker-controlled.

Study-note section WEB29

> If the server trusts it alone, authorization can be bypassed.

Study-note section WEB29

> HTTPS does not make client-generated claims trustworthy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 59 | lines 4-7

> Create a Cookie with JavaScript
> JavaScript can create, read, and delete cookies with the document.cookie property.
> With JavaScript, a cookie can be created like this:
> document.cookie = "username=John Doe";

Web Programming module | PDF page 76 | lines 2-5

> in one application. A PHP session allows you to store user information on the server for later use (i.e., username, shopping items, etc.). However, session information is temporary and will be deleted after the user has left the website. If you need a permanent storage you may want to store the data in a database.

Web Programming module | PDF page 76 | lines 17-23

> Storing a Session Variable
> The correct way to store and retrieve session variables is to use the PHP $_SESSION variable:

```
<?php
session_start();
// store session data
$_SESSION['views']=1;
```

Question sources

Computer Science Exit Exam · 2018 · Question 93

Question 45 | Q00985

Which one of the following is not feature of dynamic web page?

A. Contents can be generated on-the-fly

B. Changing content or lively

C. Contain the same prebuilt content each time the page is loaded

D. Ability to connect to a database

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: Always serving the same prebuilt content is the static-generation description.

WEB02 contrasts a basic static server returning stored files with dynamic processing constructing a representation from logic, requests, or data. C describes the prebuilt static delivery model rather than the distinctive feature of dynamic generation. Dynamic pages may still happen to display identical text across loads; the exclusion concerns same prebuilt content as a model, not a requirement that every dynamic response visibly change.

Why each choice is right or wrong

A - Incorrect: Generating content on demand is directly consistent with dynamic processing from logic, requests, or data.

B - Incorrect: Dynamic content can change with inputs or application data, although visible changes are not mandatory on every load.

C - Correct: Always returning the same prebuilt content describes the static delivery model rather than dynamic response generation.

D - Incorrect: Server-side applications can access databases while constructing dynamic responses, as described in WEB01.

Primary note proof

Study-note section WEB02

> A basic static server returns stored files; dynamic processing constructs a representation from logic, requests or data.

Study-note section WEB02

> Static pages can still run interactive browser scripts, and dynamic generation need not change visible text every time.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 8-16

> scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.
> Client-side scripting is the glamorous, eye-catching part of Web development. In contrast, server-side scripting is invisible to the user. Server-side Web scripting is mostly about connecting Web sites to back-end servers, such as databases. This enables the following types of two-way communication:
> ➢ Server to client: Web pages can be assembled from back end-server output.
> ➢ Client to server: Customer-entered information can be acted upon.

Question sources

Computer Science Exit Exam · 2018 · Question 98

Question 46 | Q00986

Why should form <input> elements include an associated <label> with the for attribute?

A. It enables offline form storage

B. It encrypts user data

C. .
It increases form submission speed

D. It improves accessibility: screen readers announce the label when the input is focused

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer D: An associated label gives the input an accessible name.

WEB09 connects a label's for value to the control's id. WEB07 requires programmatic labels for accessibility. Together these support an input label that assistive technology can announce when the control is focused. That association does not store the form offline, encrypt data, or increase submission speed; its purpose here is accessible identification of the control.

Why each choice is right or wrong

A - Incorrect: Label association identifies a control and does not create offline form storage.

B - Incorrect: A label does not encrypt values or change transport security; encryption needs a separate mechanism.

C - Incorrect: Linking a label to an input does not increase network or form-submission speed.

D - Correct: The associated label provides accessible control identification that screen readers can announce on focus.

Primary note proof

Study-note section WEB09

> A label's for matches the control's id; name identifies its submitted field.

Study-note section WEB07

> Give controls programmatic labels, clear link text and keyboard access; visual styling alone does not establish accessibility.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 21 | lines 13-22

> 2.2.10 HTML Form and Form Controls
> The <form> HTML element represents a document section containing interactive controls for submitting information. An HTML form is a section of a document containing normal content, markup, special elements called controls (checkboxes, radio buttons, menus, etc.) and labels on those controls. HTML Forms are required, when we want to collect data from a user. Using forms, we can build a website with user registration. We take the information collected from the user and then post it to the backend application such as php.
> Users interact with forms through named controls. There are various from controls available like text fields, text area, check box, drop-down menus etc.

Question sources

Computer Science Exit Exam · 2018 · Question 99

Question 47 | Q00992

Which HTML tag is used for program listing?

A. <p>

B. <pre>

C. <head>

D. <sub>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer B: <pre> preserves program-listing whitespace.

WEB06 says <pre> preserves source whitespace, which is needed for displaying an indented program listing. <code> can additionally mark code semantics, but it is not offered here. A paragraph does not preserve the listing's whitespace by itself, head is metadata, and sub marks a subscript. Therefore <pre> is the matching listed tag.

Why each choice is right or wrong

A - Incorrect: A paragraph marks prose structure and does not by itself preserve a program listing's indentation and line breaks.

B - Correct: The preformatted element preserves source whitespace, making it suitable for an indented program listing.

C - Incorrect: The head holds metadata rather than visible program-listing content.

D - Incorrect: The sub element marks subscript text and does not provide preformatted listing whitespace.

Primary note proof

Study-note section WEB06

> <pre> preserves source whitespace; <code> marks code.

Study-note section WEB06

> <!DOCTYPE html> selects modern standards mode; <html> is the document root, <head> holds metadata, and <body> contains document content.

Study-note section WEB06

> <sub>/<sup> mark sub/superscripts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 15 | lines 27-34

> Paragraph: definition of paragraph
> Format: ...
> E.g., this is a paragraph of text. it has a new line before and after it.
> The browser inserts a new line before and after the text in the paragraph tag.
> Attribute:
> align="alignment" {left, right, center, justify}
> • line break - inserts a new line.
> Format: <br>. E.g., Line one <br> line two <br> line three <br> line four

Web Programming module | PDF page 12 | lines 6-12

<head>...</head>
- Contains information which is not displayed in the browser display area and may contain other tags in it
- Both the start and end tags may be omitted and inferred from child elements (HTML5)
- The head element may contain other elements such as base, link, meta, style, script, title

Web Programming module | PDF page 16 | lines 9-12

sub/sup - define either a subscript or a superscript
Format: _{...} ; ^{...}
E.g.
X₁² + 2X₃

Question sources

National Exit Exam practice questions · Question 6

Computer Science Exit Exam practice collection · Question 30

Question 48 | Q00996

Which attribute of a text input limits the number of characters the user may enter, rather than its visible width?

A. MAXLENGTH

B. TYPE

C. ACTION

D. STYLE

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer A: MAXLENGTH

For a text input, maxlength limits entered text length, while size controls approximate visible width. The revised question removes the scrolling and new-line language because those are display concerns rather than the entered-character bound. TYPE chooses the control kind, ACTION identifies a form-processing destination and STYLE supplies presentation declarations. Browser length limits improve input behavior but must still be checked server-side when enforcing a security requirement.

Why each choice is right or wrong

A - Correct: MAXLENGTH supplies the entered-text length limit.

B - Incorrect: TYPE selects the input kind rather than a maximum length.

C - Incorrect: ACTION belongs to form destination configuration.

D - Incorrect: STYLE controls presentation rather than this input-length attribute.

Primary note proof

Study-note section WEB09

> For a text input, size indicates approximate visible width, whereas maxlength limits entered text length.

Study-note section WEB09

> A form’s action identifies its processing URL, method its submission method, and enctype its encoding.

Study-note section WEB11

> An external sheet uses <link rel="stylesheet" href="styles.css"> in head; an internal sheet uses style; an inline declaration uses an element’s style attribute.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 22 | lines 2-9

> 1. Text inputs create using the <input> tag, an input control can take many forms
> using the type attribute. Possible values for type attribute
> - Text - most common used for text inputs
> - Password - for password fields
> - Checkbox
> - Radio
> Other attributes for <input> tag
> - Name, value, size, maxlength

Web Programming module | PDF page 21 | lines 23-27

> Form attributes:
> - action → Backend script ready to process the data collected
> - method → method used to upload Data. The most frequently used
> - target → Specify the target window or frame where the result of the script
> will be displayed. It takes values like _blank, _self, _parent etc.

Question sources

National Exit Exam practice questions · Question 10

Computer Science Exit Exam practice collection · Question 31

Question 49 | Q01068

Which of the following is not JavaScript Data Types?

A. Undefined

B. Number

C. Boolean

D. Float

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer D: Float is not a separate JavaScript primitive type.

WEB16 lists undefined, null, Boolean, Number, BigInt, String, and Symbol as primitives and explicitly states that Float is not a separate Number primitive type. Floating-point numeric values belong to Number. Undefined, Number, and Boolean are therefore valid type names in the offered set, leaving Float as the exclusion.

Why each choice is right or wrong

A - Incorrect: Undefined is explicitly listed as a JavaScript primitive type in WEB16.

B - Incorrect: Number is the JavaScript numeric primitive type that includes ordinary floating-point numeric values.

C - Incorrect: Boolean is explicitly listed as the primitive type for true and false values.

D - Correct: Float is not a separate JavaScript primitive type; ordinary floating-point values use Number.

Primary note proof

Study-note section WEB16

> JavaScript primitives are undefined, null, Boolean, Number, BigInt, String and Symbol; objects include arrays, functions and dates.

Study-note section WEB16

> Float is not a separate Number primitive type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 47 | lines 18-33

> 4.5.1 JavaScript Data types
> A value in JavaScript is always of a certain type. In JavaScript, a variable can take one type at one moment and then another. This is called dynamic typing.
> - Number → represents integers and floating points. There are many operations we can perform on a number multiplication *, division /,

addition +, subtraction - and so on
- String → a string in JavaScript must be surrounded by quotes. We have three different quotes
1. Single quotes --- ‘single’
2. Double quotes --- ‘double’
3. Backticks -- `hello`. In a standard US keyboard, you can find the backticks above the tab key before the number 1 key.
Backticks allows to embed variables and expressions into a string by wrapping them with ${}
alert(`the result is ${1 + 2}`);
- Boolean → a boolean has two values true or false

Web Programming module | PDF page 48 | lines 2-5

- Objects → are used to store a key/value pair property. Objects are created using curly braces {}
- Arrays → are regular objects and used to store indexed values. Arrays are created by using square brackets []

Question sources

Web Programming practice questions · Question 1

Question 50 | Q01069

Which company developed JavaScript?

A. Netscape

B. Bell Labs

C. Sun Microsystems

D. IBM

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer A: Netscape

WEB15 explicitly states that JavaScript originated at Netscape and distinguishes it from Java. Netscape is therefore the company named by the supplied note. None of Bell Labs, Sun Microsystems, or IBM is the origin identified for JavaScript. The Java distinction is especially relevant because the similar language names can encourage a wrong company association.

Why each choice is right or wrong

A - Correct: Netscape is explicitly identified in WEB15 as the company where JavaScript originated.

B - Incorrect: Bell Labs is not the JavaScript origin given in the complete study note.

C - Incorrect: The note distinguishes JavaScript from Java and identifies Netscape, not Sun Microsystems, as its origin.

D - Incorrect: IBM is not the company identified as JavaScript's origin in WEB15.

Primary note proof

Study-note section WEB15

> JavaScript originated at Netscape; it is distinct from Java and supports sophisticated applications through prototype-based objects and first-class functions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 3-7

> 4.1 Introduction to JavaScript
> JavaScript (JS) is a lightweight, interpreted or compiled programming language. It is a single threaded, dynamic language, supporting object oriented, functional programming styles. Do not confuse JavaScript with the Java Programming language. The two-programming language have very different syntax, semantics, and use.

Question sources

Web Programming practice questions · Question 2

Question 51 | Q01070

Inside which HTML element do we put the JavaScript?

A. <script>

B. <head>

C. <meta>

D. <style>

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer A: <script>

Page 44 says JavaScript uses the script tag, and p.45 shows the enclosed script block. Evidence: PDF p. 44, PDF p. 45, PDF p. 28.

Why each choice is right or wrong

A - Correct: Actual code-containing element.

B - Incorrect: A possible surrounding document section, not the script element.

C - Incorrect: Metadata element, not code container.

D - Incorrect: Embedded stylesheet element, not JavaScript container.

Primary note proof

Study-note section WEB15

> Use <script src="app.js"></script> for an external .js file; script content has no HTML closing-tag syntax inside its JavaScript source.

Study-note section WEB06

> <!DOCTYPE html> selects modern standards mode; <html> is the document root, <head> holds metadata, and <body> contains document content.

Study-note section WEB11

> An external sheet uses <link rel="stylesheet" href="styles.css"> in head; an internal sheet uses style; an inline declaration uses an element's style attribute.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 30-33

> 4.2.1 JavaScript Syntax
> JavaScript is added to an HTML document in a similar fashion as that of CSS. The main difference is we use <script> tag instead of <link> tag.
> Here is the basic JavaScript syntax

Web Programming module | PDF page 45 | lines 2-8

> <script>
> // JavaScript code goes here
> </script>
> - Internal JavaScript is one way of including JavaScript using the script tag inside our HTML document. We can use the above syntax to include it anywhere in our document. The best place to include a JavaScript file is before the end of body tag </body>.

Web Programming module | PDF page 28 | lines 12-14

> 2. Embedded stylesheet
> This type of stylesheet should be used when a single document has a unique style. You define internal styles in the head section by using the <style> tag.

Question sources

Web Programming practice questions · Question 3

Question 52 | Q01071

Which of the following is correct about features of JavaScript?

A. It can not Handling dates and time.

B. JavaScript is a object-based scripting language.

C. JavaScript is not interpreter based scripting language.

D. All of the above

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer B: JavaScript supports prototype-based objects.

WEB15 says JavaScript supports sophisticated applications through prototype-based objects and first-class functions, directly supporting B. WEB16 also includes dates among objects, refuting the claimed inability to handle dates and time. The broad denial of interpreted scripting is not a supported language definition. Since at least A is false, All cannot hold; B is the supported positive feature.

Why each choice is right or wrong

A - Incorrect: The note includes Date objects, so the claim that JavaScript cannot handle dates and time is false.

B - Correct: Prototype-based objects support the description of JavaScript as an object-based scripting language.

C - Incorrect: This blanket denial is not a supported feature definition; browser JavaScript runs as scripts through its runtime.

D - Incorrect: All fails because the date/time incapability claim is false, while the object-based feature is supported.

Primary note proof

Study-note section WEB15

> JavaScript originated at Netscape; it is distinct from Java and supports sophisticated applications through prototype-based objects and first-class functions.

Study-note section WEB16

> JavaScript primitives are undefined, null, Boolean, Number, BigInt, String and Symbol; objects include arrays, functions and dates.

Study-note section WEB-S05

> JavaScript implementations may interpret or compile source; interpretation and compilation are execution strategies rather than competing language identities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 3-7

> 4.1 Introduction to JavaScript
> JavaScript (JS) is a lightweight, interpreted or compiled programming language. It is a single threaded, dynamic language, supporting object oriented, functional programming styles. Do not confuse JavaScript with the Java Programming language. The two-programming language have very different syntax, semantics, and use.

Question sources

Web Programming practice questions · Question 4

Question 53 | Q01072

Choose the correct JavaScript syntax to change the content of the following HTML code.

A. document.getElement ("letsfindcourse").innerHTML = "I am a letsfindcourse";

B. document.getElementById ("letsfindcourse").innerHTML = "I am a letsfindcourse";

C. document.getId ("letsfindcourse") = "I am a letsfindcourse";

D. document.getElementById ("letsfindcourse").innerHTML = I am a letsfindcourse;

Hint

Track which node is selected or created, and distinguish changing content from attaching a node.

Answer B: Use document.getElementById(...).innerHTML with a quoted string.

WEB20 identifies document.getElementById as the ID lookup and innerHTML as a descendant-markup replacement property. B combines the correct method spelling with a quoted string value. A and C invent lookup names; D lacks string quotes and cannot parse as the intended assignment. Execution still requires that the element exist, which should be checked in real code.

Why each choice is right or wrong

A - Incorrect: document.getElement is not the ID lookup method defined in WEB20.

B - Correct: The exact ID lookup and quoted innerHTML assignment are valid, assuming that the requested element exists.

C - Incorrect: document.getId is not the DOM method, and assigning directly to that call does not set element content.

D - Incorrect: The intended replacement text is unquoted, so the JavaScript assignment is syntactically invalid.

Primary note proof

Study-note section WEB20

document.getElementById("x") returns the matching element or null; querySelector(".item") returns the first match, and querySelectorAll returns a static NodeList.

Study-note section WEB20

innerHTML parses/replaces descendants, so += reparses/replaces the prior descendant content too and can destroy listeners/state.

Study-note section WEB16

Strings use either quote style, with appropriate escaping.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 52 | lines 10-20

4.10.1 Accessing HTML elements in JavaScript
In the DOM Api, there are different methods to get access to HTML elements. We will list the available methods here, but only the most recent methods are covered here.
- getElementById(id) returns an element with that specific ID
- getElmentsByClassName(class) returns elements with that specific class
getElementsByTagName(tag) returns elements with the given tag name
- querySelector(selector) → The Document method querySelector() returns the first Element within the document that matches the specified selector, or group of selectors. If no matches are found, null is returned.
element = document.querySelector(selectors);

Web Programming module | PDF page 47 | lines 24-28

- String → a string in JavaScript must be surrounded by quotes. We have three different quotes
1. Single quotes --- ‘single’
2. Double quotes --- ‘double’
3. Backticks -- `hello`. In a standard US keyboard, you can find the

Question sources

Web Programming practice questions · Question 5

Question 54 | Q01073

which of the following are advantages of JavaScript?

A. Less server interaction

B. Increased interactivity

C. Richer interfaces

D. All of the above

Hint

Separate event dispatch, handler registration and cancellation of the browser default action.

Answer D: All three are possible JavaScript benefits.

WEB01 assigns JavaScript programmable interactivity, and WEB21 describes browser checks that give immediate feedback. Such client work can avoid some server round trips, support richer controls, and increase interactivity. Each preceding option names a possible benefit rather than a guarantee of good implementation. The combined All choice therefore captures the full supported set.

Why each choice is right or wrong

A - Incorrect: Local client computation can avoid some requests, but this option states only one of the listed benefits.

B - Incorrect: JavaScript supplies programmable interactivity, but this is only part of the full supported set.

C - Incorrect: Programmable controls can provide richer interfaces, alongside the other possible benefits.

D - Correct: Reduced requests for local work, interactivity, and richer interfaces are all possible benefits of browser JavaScript.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB21

> Browser checks improve feedback; repeat validation and authorization on the server.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 9-19

> JavaScript is a scripting or programming language that allows us to implement complex features on web pages. JavaScript will enable us to display timely content updates, interactive maps, 2D/3D animated graphics etc. JavaScript is one of the three layers of web technologies which lets to add functionality to a web page. The three layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'
> What can JavaScript really do? It can
> - Stores values inside variables
> - Perform operations on pieces of texts
> - Run a code in response to events

Question sources

Web Programming practice questions · Question 6

Question 55 | Q01074

Which statement is true about modern JavaScript and browser capabilities?

A. Browser scripts cannot freely read or write arbitrary client files without an appropriate API and permission

B. JavaScript cannot be used for networking applications

C. JavaScript doesn't have any multi-threading or multiprocessor capabilities

D. All of the above

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer A: Browser scripts cannot freely read or write arbitrary client files without an appropriate API and permission

Modern JavaScript can make network requests and can use workers for concurrent computation. Browser scripts may also access selected files or devices through permission-based APIs, so their restriction is on unrestricted access, not all file interaction. The first choice states that boundary precisely. The two remaining capability denials are false, which also makes All of the above false. The answer distinguishes host permissions from the expressive power of the language.

Why each choice is right or wrong

A - Correct: The sandbox does not allow arbitrary unrestricted file access.

B - Incorrect: Browser network APIs and server runtimes refute this denial.

C - Incorrect: Workers provide concurrent computation, refuting this absolute denial.

D - Incorrect: The networking and concurrency denials are false.

Primary note proof

Study-note section WEB01

> Browser scripts are sandboxed, but may access selected files/devices through permission-based APIs.

Study-note section WEB22

> Browser workers support concurrent computation, but DOM access normally remains on the page’s main thread.

Study-note section WEB34

> With fetch, await fetch(url) obtains a response, then check response.ok and parse await response.json().

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 3-7

> 4.1 Introduction to JavaScript
> JavaScript (JS) is a lightweight, interpreted or compiled programming language. It is a single threaded, dynamic language, supporting object oriented, functional programming styles. Do not confuse JavaScript with the Java Programming language. The two-programming language have very different syntax, semantics, and use.

Web Programming module | PDF page 92 | lines 6-14

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.
> Benefits of using Node
> 1. Performance
> 2. Uses plain old JavaScript
> 3. Thousands of reusable packages
> 4. Portable

Question sources

Web Programming practice questions · Question 7

Question 56 | Q01075

Microsoft has developed a popular HTML editor called?

A. Dreamweaver

B. FrontPage

C. HomeSite

D. Macromedia

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: FrontPage

WEB02 explicitly identifies Microsoft FrontPage and Macromedia HomeSite5 as historical editors. Therefore FrontPage is the Microsoft editor requested. HomeSite belongs to the other attribution given in the note; Macromedia is a company name rather than the Microsoft product. This is historical product recognition, not a current editor recommendation.

Why each choice is right or wrong

A - Incorrect: Dreamweaver is not the Microsoft editor identified in the supplied historical attribution.

B - Correct: WEB02 explicitly identifies FrontPage as the historical Microsoft editor.

C - Incorrect: HomeSite5 is attributed to Macromedia rather than Microsoft in the note.

D - Incorrect: Macromedia is the company in the HomeSite attribution, not the requested Microsoft editor name.

Primary note proof

Study-note section WEB02

> Historical editor-name items refer to Microsoft FrontPage and Macromedia HomeSite5, not languages or required modern tools.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 10 | lines 22-26

> HTML is composed of several tags. HTML Document:-
> - A text document containing markup tags
> - The tags tell the browser how to display the document
> - Should have an .htm or .html file name extension
> - Can be created using a simple text editor

Question sources

Web Programming practice questions · Question 8

Question 57 | Q01076

HomeSite 5 is a well-liked HTML and JavaScript editor from Macromedia.

A. True

B. False

C. Can be true or false

D. Cannot say

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: True as a historical product attribution.

WEB02 explicitly identifies Macromedia HomeSite5 as a historical editor. That supports the product-and-company attribution in the statement. Well-liked is an informal popularity description, not a measurable language or browser property supplied here. On the factual historical attribution being tested, True is the supported choice; the note does not recommend the product as a current required editor.

Why each choice is right or wrong

A - Correct: The historical attribution to Macromedia HomeSite5 is explicitly supported by WEB02.

B - Incorrect: False contradicts the product-and-company attribution given in the complete note.

C - Incorrect: The historical attribution is settled in the note rather than varying between true and false.

D - Incorrect: The note supplies the historical attribution needed to answer the factual part of the statement.

Primary note proof

Study-note section WEB02

> Historical editor-name items refer to Microsoft FrontPage and Macromedia HomeSite5, not languages or required modern tools.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 10 | lines 22-26

> HTML is composed of several tags. HTML Document:-
> - A text document containing markup tags
> - The tags tell the browser how to display the document
> - Should have an .htm or .html file name extension
> - Can be created using a simple text editor

Question sources

Web Programming practice questions · Question 9

Question 58 | Q01077

Outside strings and comments, in a context where token boundaries are preserved and line breaks cannot trigger different parsing or automatic semicolon insertion, which single option gives the complete selection of whitespace that can be ignored?

A. spaces

B. tabs

C. newlines

D. All of the above

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer D: All of the above

JavaScript normally uses whitespace to separate tokens. In the explicitly restricted context, extra spaces, tabs and newlines can all be removed without changing the program's meaning. The exclusions matter: whitespace inside strings is data, and line breaks can change parsing in other contexts. All of the above is therefore the complete single choice here; each individual whitespace kind is eligible but would be an incomplete selection.

Why each choice is right or wrong

A - Incorrect: Spaces qualify here, but this choice omits the other qualifying whitespace.

B - Incorrect: Tabs qualify here, but this choice is incomplete.

C - Incorrect: Newlines qualify under the explicit parsing restriction, but this choice is incomplete.

D - Correct: This includes all three qualifying whitespace kinds in the stated context.

Primary note proof

Study-note section WEB15

> Whitespace usually separates tokens but newlines can affect parsing, especially after return.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 45 | lines 2-11

> <script>
> // JavaScript code goes here
> </script>
> - Internal JavaScript is one way of including JavaScript using the script tag inside our HTML document. We can use the above syntax to include it anywhere in our document. The best place to include a JavaScript file is before the end of body tag </body>.
> - <script>

```
console.log(“Hello world”)
</script>
```

Question sources

Web Programming practice questions · Question 10

Question 59 | Q01078

JavaScript is designed for following purpose -

A. to style HTML pages

B. to execute Queries related to databases on a server

C. to add interactivity to html pages

D. All of the above

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: to add interactivity to html pages

The module assigns JavaScript interactive features/functionality, HTML structure and CSS presentation (p.44). Evidence: PDF p. 44, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: Styling is principally CSS in this division.

B - Incorrect: Server database query execution is not the stated browser JavaScript purpose.

C - Correct: Matches interactive feature role.

D - Incorrect: The module role distinction prevents all three being its stated primary design purpose.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 9-19

> JavaScript is a scripting or programming language that allows us to implement complex features on web pages. JavaScript will enable us to display timely content updates, interactive maps, 2D/3D animated graphics etc. JavaScript is one of the three layers of web technologies which lets to add functionality to a web page. The three layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'
> What can JavaScript really do? It can
> - Stores values inside variables
> - Perform operations on pieces of texts
> - Run a code in response to events

Web Programming module | PDF page 60 | lines 11-16

> Client-side scripting is the glamorous, eye-catching part of Web development. In contrast, server-side scripting is invisible to the user. Server-side Web scripting is mostly about connecting Web sites to back-end servers, such as databases. This enables the following types of two-way communication:
> ➢ Server to client: Web pages can be assembled from back end-server output.
> ➢ Client to server: Customer-entered information can be acted upon.

Question sources

Web Programming practice questions · Question 11

Question 60 | Q01079

What will be the output of the following JavaScript code?
var string1 = "Letsfindcourse"; var intvalue = 30; alert (string1 + intvalue);

A. Letsfindcourse 30

B. 30

C. Letsfindcourse30

D. Exception

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer C: Letsfindcourse30

WEB16 states that + can concatenate when a string is involved. Here string1 is Letsfindcourse and intvalue is number 30, so the numeric operand is represented as text and appended to the string. Neither operand contains a trailing or leading space. The alert therefore displays Letsfindcourse30, not the spaced version.

Why each choice is right or wrong

A - Incorrect: The code contains no separator space between the string and number, so this adds an unwritten character.

B - Incorrect: The original string is retained by concatenation rather than discarded.

C - Correct: The string operand triggers concatenation, appending 30 directly with no extra space.

D - Incorrect: String-plus-number concatenation is valid JavaScript and does not throw merely because the operand types differ.

Primary note proof

Study-note section WEB16

> + can concatenate when a string is involved; arithmetic / converts ordinary numeric strings.

Study-note section WEB22

> alert displays information, confirm returns a Boolean, and prompt requests text or returns null on cancellation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 47 | lines 21-28

> - Number → represents integers and floating points. There are many
> operations we can perform on a number multiplication *, division /,
> addition +, subtraction - and so on
> - String → a string in JavaScript must be surrounded by quotes. We have
> three different quotes
> 1. Single quotes --- ‘single’
> 2. Double quotes --- ‘double’
> 3. Backticks -- `hello`. In a standard US keyboard, you can find the

Web Programming module | PDF page 46 | lines 19-20

> var name = prompt("Enter your name: ");
> alert("Hello " + name);

Web Programming module | PDF page 48 | lines 23-33

> 4.5.3 Data Type Conversion
> Type conversations are a very common occurrence in JavaScript. Operators and
> functions converts the values given to them to the right type. Mathematical operations
> convert value to numbers.
> - String conversion → we can use String(value) function to convert a value
> to string
> - Numeric Conversion → happens automatically in mathematical operations
> and expression. We can use the Number(value) function to explicitly
> convert a value to a number:

```
let str = "123";
let num = Number(str);
```

Question sources

Web Programming practice questions · Question 12

Question 61 | Q01080

Among the following, which one is a ternary operator in JavaScript?

A. #

B. ::

C. &:

D. ?:

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer D: ?:

The module explicitly gives condition ? value1 : value2 as the ternary operator syntax (p.50). Evidence: PDF p. 50.

Why each choice is right or wrong

A - Incorrect: Not the shown syntax.

B - Incorrect: Double colon is not the shown syntax.

C - Incorrect: Ampersand-colon is not the shown syntax.

D - Correct: Question mark and colon match the shown conditional expression.

Primary note proof

Study-note section WEB17

> A condition ? x : y chooses one branch.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 50 | lines 15-18

> Ternary Operators:
> Ternary operator is represented by a question mark “?”. We can use ternary
> operator to perform conditional operations like if-else statement
> let result = condition ? value1 : value2;

Question sources

Web Programming practice questions · Question 13

Question 62 | Q01081

Which option names the three clauses of the general JavaScript for-loop form, regardless of how the loop variable changes?

A. the initialization, the Incrementation, and update

B. the initialization, the test, and the update

C. the initialization, the test, and Incrementation

D. All of the above

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: the initialization, the test, and the update

The general form is for(initialization; test; update). Initialization runs once, the test decides whether another iteration starts, and the update follows each completed body. An update can increment, decrement or perform another expression. Therefore an increment-only description is narrower than the general clause name. The first alternative also omits the test, so the aggregate alternative is incorrect.

Why each choice is right or wrong

A - Incorrect: Incrementation and update overlap, and the condition/test clause is missing.

B - Correct: These are the three general clause names, independent of the particular update expression.

C - Incorrect: Incrementing is one possible update; it is not the general name for a clause that may instead decrement or do something else.

D - Incorrect: The first alternative omits the test, and the increment-only alternative is not the general form.

Primary note proof

Study-note section WEB19

> for(initialization;test;update) initializes once, checks before each iteration, then updates after each body.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 50 | lines 30-33

> - For loop → syntax
> for (begin; condition; step) {
> // ... loop body ...
> }

Question sources

Web Programming practice questions · Question 14

Question 63 | Q01082

Which JavaScript operators compare equality and inequality without type coercion, while == and != remain available for coercive comparison?

A. It uses bitwise checking

B. === and !==

C. It uses equals() and notequals() instead

D. It uses equalto()

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer B: === and !==

JavaScript supports both loose and strict equality operators. == and != can coerce types, while === and !== compare without that coercion. The strict operators are alternatives with different comparison rules rather than replacements that remove the loose operators from the language. Bitwise checking and the invented equality function names do not identify these language operators, so the corrected question has a single supported answer.

Why each choice is right or wrong

A - Incorrect: Bitwise operations are different from strict equality comparisons.

B - Correct: These are the strict equality and inequality operators.

C - Incorrect: These function names are not the language's strict comparison operators.

D - Incorrect: This is not the language's strict equality operator.

Primary note proof

Study-note section WEB17

> JavaScript has ==/!= with coercion and ===/!== without coercion; strict operators do not replace or remove the loose ones.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 49 | lines 2-8

> 4.6 Arithmetic and Logical Operators in JavaScript
> Arithmetic operators are used to perform mathematical operations between numeric operands. We can perform different operations on numbers, like addition, subtraction multiplication. Here is the list of arithmetic operators we can use on numbers

Logical Operators:
In JavaScript, the logical operators are used to combine two or more conditions.
JavaScript provides the following logical operators.

Question sources

Web Programming practice questions · Question 15

Question 64 | Q01083

What should appear at the very end of your JavaScript?The <script language="javascript"> tag

A. <script>

B. </script>

C. </script language="javascript">

D. All of the above

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B: </script>

The script block shown on p.45 ends with </script>; attributes belong in the opening tag.
Evidence: PDF p. 45.

Why each choice is right or wrong

A - Incorrect: Opening tag does not close the block.

B - Correct: Exact closing delimiter.

C - Incorrect: The demonstrated closing tag has no attributes.

D - Incorrect: Only B matches the actual closing delimiter.

Primary note proof

Study-note section WEB15

> Use <script src="app.js"></script> for an external .js file; script content has no HTML closing-tag syntax inside its JavaScript source.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 45 | lines 2-4

```
<script>
// JavaScript code goes here
</script>
```

Question sources

Web Programming practice questions · Question 16

Question 65 | Q01084

Which listed item is a directive string rather than a JavaScript keyword that introduces a statement form? Treat with as its legacy non-strict form.

A. if

B. with

C. debugger

D. use strict

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer D: use strict

A quoted "use strict" in a directive prologue selects strict semantics. It is not a reserved keyword introducing a special statement form, although the quoted text is syntactically an expression statement. By contrast, if introduces a conditional statement, with names a legacy scope-lookup statement forbidden in strict mode, and debugger names a statement that can invoke a debugger facility.

Why each choice is right or wrong

A - Incorrect: if names the conditional statement form, so it is not the requested non-keyword directive.

B - Incorrect: with names a statement form, even though strict mode restricts its use.

C - Incorrect: debugger names a JavaScript statement form rather than a directive string.

D - Correct: "use strict" is a directive string, not a reserved statement keyword; technically its quoted form is an expression statement.

Primary note proof

Study-note section WEB15

> “use strict” is a directive string, not a reserved statement keyword.

Study-note section WEB-S05

> An if statement conditionally executes code.

Study-note section WEB-S05

> The legacy with statement temporarily extends name lookup and is forbidden in strict mode.

Study-note section WEB-S05

> A debugger statement may invoke an active debugging facility.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 49 | lines 9-14

> 4.7 Control Structures (Conditional and Looping Statements)
> The “if” statement:
> The if(...) statement evaluates a condition in parentheses and, if the result is true, executes a block of code.
> let age = 20;
> if (age > 18) alert('You are right!');

Question sources

Web Programming practice questions · Question 17

Question 66 | Q01086

Which symbol is used for comments in JavaScript?

A. \\

B. //

C. * *\

D. * */

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B: //

The comment rule explicitly uses // for a single line and /* */ for multiline (p.46). All alternatives except B reverse slash direction. Evidence: PDF p. 46.

Why each choice is right or wrong

A - Incorrect: Backslashes are not the rule.

B - Correct: Exact single-line delimiter.

C - Incorrect: Backslash-star is not forward-slash-star.

D - Incorrect: Incorrect opening backslash, even though closing portion resembles a block end.

Primary note proof

Study-note section WEB15

> // comments end at line end; /* */ span a block.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 46 | lines 4-14

4.3 JavaScript Comments
As with HTML and CSS, it is possible to write comments into a JavaScript code that will be ignored by the browser, and exist to provide instructions to fellow developers on how the code works. There are two types of JavaScript comments
- A single line comment written after a double forward slash (//)
// I am a single line comment
- A multiline comment is written between the strings (/* */)
/*
I am
a multi line comment
*/

Question sources

Web Programming practice questions · Question 19

Question 67 | Q01087

Which of them is not the looping structures in JavaScript?

A. for

B. while

C. forwhich

D. dowhile

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: forwhich is not a JavaScript loop form.

WEB19 defines for, while, and do...while loops. The option dowhile names that last construct informally; its actual source spelling is do ... while. forwhich is not one of the language's loop forms. Thus C is the intended non-loop name without treating informal collapsed words as executable source tokens.

Why each choice is right or wrong

A - Incorrect: for is explicitly a JavaScript loop form with initialization, test, and update clauses.

B - Incorrect: while is explicitly a loop that tests before its body.

C - Correct: forwhich is not a defined JavaScript loop form in the note.

D - Incorrect: Read as the loop name do...while, this is the supported at-least-once construct, not a literal token.

Primary note proof

Study-note section WEB19

for(initialization;test;update) initializes once, checks before each iteration, then updates after each body.

Study-note section WEB19

while checks before its body; do...while executes at least once.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 50 | lines 19-33

Loops
Loops are a way to repeat the same code multiple times.
- While loop → while loop has the following syntax
while (condition) {
//"loop body"
}
- Do-while loop → do while loops are very similar to that of while loops, but
in do while even if the condition is false there is always on execution
do{
//"loop body"
} while (condition);
- For loop → syntax
for (begin; condition; step) {
// ... loop body ...
}

Question sources

Web Programming practice questions · Question 20

Question 68 | Q01088

Which operations evaluate to undefined in ordinary JavaScript? A variable read means a direct expression read, not typeof. Select all that apply.

A. Directly reading an undeclared variable

B. Reading a declared var variable before it has been assigned a value

C. Reading a missing property of an ordinary object

D. All of the above

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer B, C: Reading a declared var variable before it has been assigned a value; Reading a missing property of an ordinary object

An unassigned declared var variable has the value undefined, and an absent ordinary object property also evaluates to undefined. A direct read of an undeclared variable instead normally throws ReferenceError. The question excludes typeof and uses var to avoid a block-scoped temporal-dead-zone case. Both B and C must be selected; All of the above is false because it would include the undeclared-variable read.

Why each choice is right or wrong

A - Incorrect: A direct undeclared-variable read throws ReferenceError.

B - Correct: An unassigned var binding evaluates to undefined.

C - Correct: An absent ordinary object property evaluates to undefined.

D - Incorrect: This includes A, which throws instead of producing undefined.

Primary note proof

Study-note section WEB16

> An unassigned declared variable or missing ordinary object property evaluates to undefined; directly reading an undeclared variable usually throws ReferenceError.

Study-note section WEB15

> var has function/global scope; let/const are block-scoped.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 6-21

> 4.5.2 Variable declaration in JavaScript
> Variables are used to store this information. A variable is a “named storage” for data. We can use variables to store goodies, visitors, and other data. We have three ways of creating variables in JavaScript
> - Using the let keyword → the following statement declares a variable
> let message;
> now we can put values using the assignment operator (=)
> - Using var keyword → we can also declare variable using the var keyword in a similar fashion as let. There is a subtle difference between declaring a variable using var and let. Variables declared with var are hoisted where as those declared with let are not. The other difference between var and let is variables declared with let has a block level scope unlike that of var. Variables declared with var have function level scope
> - Using const keyword → to declare a constant variable we can use const instead of var and let. Variables declared with const can not be changed once they are declared.

Question sources

Web Programming practice questions · Question 21

Question 69 | Q01089

What are the types of Pop up boxes available in JavaScript?

A. Alert

B. Prompt

C. Confirm

D. All of the above

Hint

Distinguish the browser object or dialog role from the kind of response it collects.

Answer D: All of the above

Prompt shows an input modal, confirm a choice modal (p.46), and alert an informational popup (p.47). Thus all three listed dialog forms are covered. Evidence: PDF p. 46, PDF p. 47.

Why each choice is right or wrong

A - Incorrect: A covered popup, but not the full list.

B - Incorrect: A covered input popup.

C - Incorrect: A covered confirmation popup.

D - Correct: Combines the three documented forms.

Primary note proof

Study-note section WEB-S03

> alert(message), confirm(message) and prompt(message) all present a supplied message.

Study-note section WEB22

> alert displays information, confirm returns a Boolean, and prompt requests text or returns null on cancellation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 46 | lines 16-25

> Accepting user input in JavaScript actually is not a difficult task. JavaScript has a function called prompt, which is used to accept input from the user. But for the result to be useful for us, it has to be stored in a variable. Here is a simple example
> var name = prompt("Enter your name: ");
> alert("Hello " + name);
> ➢ Prompt will show a modal windows with a text message, an input field for the

visitor, and the buttons OK/Cancel
➢ We can also use confirm to get information from a user:- the function confirm shows a modal window with a question and two buttons: OK and Cancel
The result is true if OK is pressed and false otherwise

Web Programming module | PDF page 47 | lines 2-2

2. Alert: shows a browser pop up with the information inside the parenthesis

Question sources

Web Programming practice questions · Question 22

Question 70 | Q01090

what is the disadvantage of using innerHTML in JavaScript?

A. Content cannot be replaced everywhere

B. We can use like "appending to innerHTML"

C. Even if you use +=like "innerHTML = innerHTML + 'html'" still the old content is replaced by html

D. The entire innerHTML content is not re-parsed and build into elements, therefore its not slower

Hint

Track which node is selected or created, and distinguish changing content from attaching a node.

Answer C: Even innerHTML += reparses and replaces descendant nodes.

WEB20 says innerHTML parses and replaces descendants. The += form reads the old markup string, concatenates new markup, and assigns the entire result, so prior descendants are recreated rather than preserved. Existing node listeners or state can be lost. C captures that disadvantage; D explicitly denies the reparsing behavior and is false.

Why each choice is right or wrong

A - Incorrect: This vague restriction is not the specific innerHTML drawback established here.

B - Incorrect: Using apparent append syntax is possible; the disadvantage is that it still replaces and reparses existing descendants.

C - Correct: The += assignment reparses and replaces prior descendant content, potentially losing node listeners and state.

D - Incorrect: The note says the entire assigned markup is reparsed; this option incorrectly denies that mechanism.

Primary note proof

Study-note section WEB20

innerHTML parses/replaces descendants, so += reparses/replaces the prior descendant content too and can destroy listeners/state.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 52 | lines 3-9

4.10 JavaScript DOM (Document object Model)
The Document Object Model (DOM) is the data representation of the objects that comprise the structure and content of a document on the web. The Document Object Model (DOM) is a programming interface for web documents. It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects; that way, programming languages can interact with the page.

Question sources

Web Programming practice questions · Question 23

Question 71 | Q01091

What are the two basic groups of dataypes in JavaScript?

A. Primitive

B. Reference types.

C. All of the above

D. None of the above

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer C: Both primitive and object/reference groups.

WEB16 lists primitive values and contrasts them with objects such as arrays, functions, and dates. In the informal two-group teaching model, those object values are often called reference types. A and B each name only one group; All combines the two categories requested. This does not make Array a separate primitive or replace the detailed primitive-type list.

Why each choice is right or wrong

A - Incorrect: Primitive names one of the two requested groups but omits the object/reference group.

B - Incorrect: Reference types informally names the object group but omits the primitive group.

C - Correct: All combines primitives and the informal object/reference group, supplying both categories requested.

D - Incorrect: The note explicitly distinguishes primitives from objects, so the two-group classification is available.

Primary note proof

Study-note section WEB16

> JavaScript primitives are undefined, null, Boolean, Number, BigInt, String and Symbol; objects include arrays, functions and dates.

Study-note section WEB16

> Arrays are objects, even though informal texts call them an array data type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 47 | lines 18-33

> 4.5.1 JavaScript Data types
> A value in JavaScript is always of a certain type. In JavaScript, a variable can take one type at one moment and then another. This is called dynamic typing.
> - Number → represents integers and floating points. There are many operations we can perform on a number multiplication *, division /, addition +, subtraction - and so on
> - String → a string in JavaScript must be surrounded by quotes. We have three different quotes
> 1. Single quotes --- 'single'
> 2. Double quotes --- 'double'
> 3. Backticks -- `hello`. In a standard US keyboard, you can find the backticks above the tab key before the number 1 key.
> Backticks allows to embed variables and expressions into a string by wrapping them with ${}
> alert(`the result is ${1 + 2}`);
> - Boolean → a boolean has two values true or false

Web Programming module | PDF page 48 | lines 2-5

> - Objects → are used to store a key/value pair property. Objects are created using curly braces {}
> - Arrays → are regular objects and used to store indexed values. Arrays are created by using square brackets []

Question sources

Web Programming practice questions · Question 24

Question 72 | Q01092

Which of the following are the errors in JavaScript?

A. Load time errors

B. Run time errors:

C. Logical Errors:

D. All of the above

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer D: All three broad error categories.

WEB19 distinguishes parse-time syntax errors, runtime thrown errors, and logical errors. Interpreting load time errors as the question's broad label for parse/load failures, all three listed categories are valid. They differ in when and how failure appears: parsing may prevent execution, runtime errors occur during execution, and logical errors can produce incorrect results without throwing.

Why each choice is right or wrong

A - Incorrect: Parse or load failures are a valid broad error category, but this option omits the other listed categories.

B - Incorrect: Thrown execution errors are runtime errors, but they are only one of the listed categories.

C - Incorrect: Logical errors can produce wrong results without throwing, but they are not the only error category.

D - Correct: All covers the broad parse/load, runtime, and logical-error categories distinguished in the note.

Primary note proof

Study-note section WEB-S05

> A syntax error prevents source from being parsed successfully.

Study-note section WEB-S05

> A runtime error arises while executing a program.

Study-note section WEB-S05

> A logical error is a defect in the algorithm or expression that can produce an incorrect result without throwing an exception.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 54 | lines 10-19

4.10.4 Handling Exception in JavaScript
There's a syntax construct try...catch that allows us to "catch" errors so the script can, instead of dying, do something more reasonable.
The "try...catch" syntax
The try...catch construct has two main blocks: try, and then catch:

```
try {
// code...
} catch (err) {
// error handling
}
```

Question sources

Web Programming practice questions · Question 25

Question 73 | Q01093

Which JavaScript function capability is illustrated by storing a function in a variable and passing it as an argument?

A. First-class functions

B. Functions restricted to class-member calls

C. Functions restricted to one fixed call site

D. All of the above

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer A: First-class functions

JavaScript functions can be stored and passed as values. This is the first-class-function capability: a function can be assigned to a variable and supplied as an argument. Functions are not restricted to class-member invocation or one fixed call site. Those restrictions conflict with the stated ability to pass a function value, so the aggregate alternative is also false.

Why each choice is right or wrong

A - Correct: Storing and passing a function as a value is the first-class-function capability.

B - Incorrect: A function value can be stored and passed outside a class-member call; JavaScript does not impose this restriction.

C - Incorrect: Passing a function value allows its use at other call sites, contradicting the proposed fixed-site restriction.

D - Incorrect: The restrictions in the other two alternatives conflict with first-class function values, so the aggregate is false.

Primary note proof

Study-note section WEB19

function f(x){return x*2;} has a parameter and returns a value; functions can be passed/stored like other values.

Study-note section WEB15

JavaScript originated at Netscape; it is distinct from Java and supports sophisticated applications through prototype-based objects and first-class functions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 4-7

JavaScript (JS) is a lightweight, interpreted or compiled programming language. It is a single threaded, dynamic language, supporting object oriented, functional programming styles. Do not confuse JavaScript with the Java Programming language. The two-programming language have very different syntax, semantics, and use.

Web Programming module | PDF page 51 | lines 18-30

4.9 JavaScript Functions
Functions are the main “building blocks” of the program. They allow the code to be called many times without repetition. To create a function, we can use a function declaration.
function showCSMessage() {
alert('Hello CS Students!');
}
To create a function, first we use the function keyword then the function name.
There are times when we remove the function name. This type Functions without a function name are called anonyms functions.
After the function name we optionally put parameters in between the parenthesis, if we don’t have parameters, we always have to put the empty parenthesis and then finally the block of code to be executed.

Question sources

Web Programming practice questions · Question 26

Question 74 | Q01094

Which exact name is not a standard screen property in a normal browser page?

A. availHeight

B. colorsDepth

C. availWidth

D. colorDepth

Hint

Distinguish the browser object or dialog role from the kind of response it collects.

Answer B: colorsDepth

JavaScript property names are case-sensitive. The standard names are availHeight, availWidth and colorDepth, all with the shown camelCase spelling. The distractor colorsDepth has an extra s and is not the standard color-depth property. The valid names use their exact case-sensitive spelling.

Why each choice is right or wrong

A - Incorrect: This is the exact available-height property name.

B - Correct: The extra s makes this different from colorDepth.

C - Incorrect: This is the exact available-width property name.

D - Incorrect: This is the exact color-depth property name.

Primary note proof

Study-note section WEB22

> screen.availHeight, screen.availWidth and screen.colorDepth have exact camelCase spelling; ColorsDepth is not the property.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 56 | lines 26-30

> Window Size
> Two properties can be used to determine the size of the browser window.
> Both properties return the sizes in pixels:
> window.innerHeight - the inner height of the browser window (in pixels)
> window.innerWidth - the inner width of the browser window (in pixels)

Question sources

Web Programming practice questions · Question 27

Question 75 | Q01095

JavaScript string using double quotes is exactly the same as a string using single quotes?

A. True

B. False

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer A: True

The module lists single and double quotes as two valid ways to surround strings (p.47). For identical content requiring no different escaping, they represent the same string value. Evidence: PDF p. 47.

Why each choice is right or wrong

A - Correct: Correct for the same content under the stated assumption.

B - Incorrect: No difference in value solely from the quote choice is described.

Primary note proof

Study-note section WEB16

> Strings use either quote style, with appropriate escaping.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 47 | lines 24-31

> - String → a string in JavaScript must be surrounded by quotes. We have
> three different quotes
> 1. Single quotes --- ‘single’
> 2. Double quotes --- ‘double’
> 3. Backticks -- `hello`. In a standard US keyboard, you can find the
> backticks above the tab key before the number 1 key.
> Backticks allows to embed variables and expressions into a string
> by wrapping them with ${}

Question sources

Web Programming practice questions · Question 28

Question 76 | Q01096

What does this JavaScript output?
var a = "20";
var b = a = 30;
document.write(a + b);

A. Error in Script

B. '20'30

C. 2030

D. 60

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer D: 60

Assignment first stores the Number 30 in a, replacing its previous string value, and then assigns that same Number to b. At the addition, both operands are therefore numeric thirty. Their sum is sixty, which document.write emits. The choice supplies the computed result rather than preserving the incorrect fifty. String concatenation does not occur because the original string no longer remains in either operand.

Why each choice is right or wrong

A - Incorrect: The assignments and addition are valid JavaScript.

B - Incorrect: The original string has been replaced and quotes are not emitted.

C - Incorrect: Neither operand is the string twenty when addition occurs.

D - Correct: Both operands are Number 30, giving the supplied numeric sum.

Primary note proof

Study-note section WEB17

> var a="20"; var b=a=30; assigns Number 30 to both, so a+b is 60, not 50.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 10-13

> - Using the let keyword → the following statement declares a variable
> let message;
> now we can put values using the assignment operator (=)
> - Using var keyword → we can also declare variable using the var keyword

Web Programming module | PDF page 47 | lines 21-23

- Number → represents integers and floating points. There are many operations we can perform on a number multiplication *, division /, addition +, subtraction - and so on

Web Programming module | PDF page 47 | lines 3-9

3. The write() method
- The write() method writes HTML expressions or JavaScript code to a document.
- The write() method is mostly used for testing: If it is used after an HTML document is fully loaded, it will delete all existing HTML.
Usage:
document.write(expr)

Question sources

Web Programming practice questions · Question 29

Question 77 | Q01097

What is divide by 0 in JavaScript?
var a = 10;var b = 0;document.write(a/b);

A. Nothing is printed

B. 0 is printed

C. Infinity is printed

D. Some Garbage Value

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer C: Infinity is printed.

WEB16 states that a nonzero Number divided by zero produces signed Infinity. Here a is positive 10 and b is positive zero, so a/b yields positive Infinity. document.write converts that value to its displayed text. This is not the 0/0 case, which produces NaN, and ordinary Number division does not yield an arbitrary garbage value.

Why each choice is right or wrong

A - Incorrect: The expression evaluates to Infinity and document.write displays its text, so output is not empty.

B - Incorrect: Dividing positive ten by zero does not produce zero under JavaScript Number arithmetic.

C - Correct: A positive nonzero Number divided by positive zero yields Infinity, which is printed.

D - Incorrect: JavaScript defines this result as Infinity rather than an unspecified garbage value.

Primary note proof

Study-note section WEB16

> "abc"/"def" produces NaN; nonzero Number divided by zero produces signed Infinity, while 0/0 produces NaN.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 47 | lines 21-23

> - Number → represents integers and floating points. There are many operations we can perform on a number multiplication *, division /, addition +, subtraction - and so on

Web Programming module | PDF page 49 | lines 2-5

> 4.6 Arithmetic and Logical Operators in JavaScript
> Arithmetic operators are used to perform mathematical operations between numeric operands. We can perform different operations on numbers, like addition, subtraction multiplication. Here is the list of arithmetic operators we can use on numbers

Question sources

Web Programming practice questions · Question 30

Question 78 | Q01098

Which of the following function of Array object applies a function simultaneously against two values of the array (from right-to-left) as to reduce it to a single value?

A. pop()

B. reduce()

C. push()

D. reduceRight()

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer D: reduceRight(), applied sequentially from right to left.

WEB18 distinguishes reduce from reduceRight by traversal direction. reduceRight combines an accumulator with each element from right to left, giving the single accumulated result described. The stem's simultaneously is inaccurate: callbacks are sequential, not parallel. Pop and push change the array's end, while reduce processes left to right. D uniquely matches the intended directional reduction.

Why each choice is right or wrong

A - Incorrect: pop removes the last item and does not repeatedly combine items into an accumulated result.

B - Incorrect: reduce combines left to right, contrary to the direction specified in the question.

C - Incorrect: push appends items at the array end rather than reducing existing items to one accumulated value.

D - Correct: reduceRight performs the requested right-to-left reduction, but sequentially rather than literally simultaneously.

Primary note proof

Study-note section WEB18

> reduce combines sequentially from left to right; reduceRight combines sequentially from right to left with an accumulator, not simultaneous parallel execution.

Study-note section WEB18

> Array push/pop add/remove at the end; unshift/shift act at the beginning.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 51 | lines 13-17

> Methods
> - Push → appends an element to the end
> - Pop → takes an element from the end.
> - Shift → Extracts the first element of the array and returns it
> - Unshift → Add the element to the beginning of the array:

Question sources

Web Programming practice questions · Question 31

Question 79 | Q01099

What is the output of following Javascript?var a = 'letsfind';var b = 'course';var c = a/b;document.write(c);

A. letsfindcourse

B. letsfind/course

C. NaN

D. None of the above

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer C: NaN

WEB16 says arithmetic division converts ordinary numeric strings and gives NaN when nonnumeric strings are divided. Neither letsfind nor course is a numeric string, so a/b cannot produce a meaningful Number and evaluates to NaN. Division does not concatenate strings or insert a slash into displayed text. document.write therefore prints NaN.

Why each choice is right or wrong

A - Incorrect: Concatenation would use + with strings; the source instead applies arithmetic division.

B - Incorrect: The slash is an operator, not a literal character appended between the two strings.

C - Correct: Both operands are nonnumeric strings, so numeric division evaluates to NaN and that text is written.

D - Incorrect: NaN is explicitly offered and matches the defined conversion result.

Primary note proof

Study-note section WEB16

> + can concatenate when a string is involved; arithmetic / converts ordinary numeric strings.

Study-note section WEB16

> "abc"/"def" produces NaN; nonzero Number divided by zero produces signed Infinity, while 0/0 produces NaN.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 23-33

> 4.5.3 Data Type Conversion
> Type conversations are a very common occurrence in JavaScript. Operators and functions converts the values given to them to the right type. Mathematical operations convert value to numbers.
> - String conversion → we can use String(value) function to convert a value to string
> - Numeric Conversion → happens automatically in mathematical operations and expression. We can use the Number(value) function to explicitly convert a value to a number:
> let str = "123";
> let num = Number(str);

Web Programming module | PDF page 49 | lines 2-5

> 4.6 Arithmetic and Logical Operators in JavaScript
> Arithmetic operators are used to perform mathematical operations between numeric operands. We can perform different operations on numbers, like addition, subtraction multiplication. Here is the list of arithmetic operators we can use on numbers

Question sources

Web Programming practice questions · Question 32

Question 80 | Q01100

What does the following JavaScript code write? var a = 1; document.write(a--); document.write(a);

A. 00

B. 01

C. 11

D. 10

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer D: 10

The source uses a-- despite the introductory mention of ++. Starting with a=1, postfix decrement yields the old value 1 to the first write and then changes a to 0. The second write outputs 0. No separator is written between calls, so the displayed result is 10, exactly as WEB17 explains.

Why each choice is right or wrong

A - Incorrect: Postfix decrement yields the original one for the first write, so the first digit cannot be zero.

B - Incorrect: This reverses the old-value and updated-value order of the two writes.

C - Incorrect: The decrement changes a to zero before the second write, so the second digit is not one.

D - Correct: The first write gets old value one, then the second gets updated zero, yielding adjacent text 10.

Primary note proof

Study-note section WEB17

> With a=1, printing a-- then a prints10.

Study-note section WEB17

> Postfix a++ yields the old value and then increments a.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 49 | lines 2-5

> 4.6 Arithmetic and Logical Operators in JavaScript
> Arithmetic operators are used to perform mathematical operations between numeric

operands. We can perform different operations on numbers, like addition, subtraction multiplication. Here is the list of arithmetic operators we can use on numbers

Web Programming module | PDF page 47 | lines 3-9

3. The write() method
- The write() method writes HTML expressions or JavaScript code to a document.
- The write() method is mostly used for testing: If it is used after an HTML document is fully loaded, it will delete all existing HTML.
Usage:
document.write(expr)

Question sources

Web Programming practice questions · Question 33

Question 81 | Q01101

Find output of below JavaScript addition codedocument.write("1 plus 1 is " + 1 + 1);

A. 2

B. 1 plus 1 is 2

C. 1 plus 1 is 11

D. 1 plus 1 is 1 + 1

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer C: 1 plus 1 is 11

WEB17 explicitly traces this expression. Starting with a string makes the first + concatenate the first numeric 1, producing a longer string. The next + then concatenates the second 1 rather than adding numbers. The original literal supplies the words and spaces, while the two appended digits have no extra separator. The result is 1 plus 1 is 11.

Why each choice is right or wrong

A - Incorrect: The initial string is retained, and the two numbers are not added together first.

B - Incorrect: This would require grouping the numeric addition before concatenating it with the initial string.

C - Correct: Left-to-right evaluation concatenates each one to the existing string, producing the final digits 11.

D - Incorrect: The plus signs are operators rather than literal characters in the output string.

Primary note proof

Study-note section WEB17

"1 plus 1 is "+1+1 concatenates left to right to "1 plus 1 is 11".

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 23-33

4.5.3 Data Type Conversion
Type conversations are a very common occurrence in JavaScript. Operators and functions converts the values given to them to the right type. Mathematical operations convert value to numbers.
- String conversion → we can use String(value) function to convert a value to string
- Numeric Conversion → happens automatically in mathematical operations and expression. We can use the Number(value) function to explicitly convert a value to a number:
let str = "123";
let num = Number(str);

Question sources

Web Programming practice questions · Question 34

Question 82 | Q01102

In JavaScript, Arrays are data type. State True or False

A. True

B. False

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer A: True

The data-type discussion explicitly includes arrays and defines them as regular objects storing indexed values. True fits the broad teaching use of data type; this does not mean arrays are a separate primitive type. Evidence: PDF p. 48.

Why each choice is right or wrong

A - Correct: Arrays are included in the type discussion as objects.

B - Incorrect: False would require the narrower unspoken meaning of separate primitive type.

Primary note proof

Study-note section WEB16

> Arrays are objects, even though informal texts call them an array data type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 2-5

> - Objects → are used to store a key/value pair property. Objects are created using curly braces {}
> - Arrays → are regular objects and used to store indexed values. Arrays are created by using square brackets []

Question sources

Web Programming practice questions · Question 35

Question 83 | Q01103

Does JavaScript allow exception handling?

A. Yes, it provides try, catch as well as throw key word for exception handling

B. Yes, but it provides only try block

C. Yes, but it provides only Try catch block, but does not allow throw exception

D. No

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Yes: try, catch, and throw are supported.

WEB19 explicitly states that try/catch handles thrown errors during execution and throw signals an error. Thus A includes both handling and signaling mechanisms. The alternatives claiming only try, denying throw, or denying exception handling contradict that rule. This concerns runtime exception handling; it does not imply that every parse-time or logical error is caught automatically.

Why each choice is right or wrong

A - Correct: JavaScript provides try/catch to handle thrown errors and throw to signal them, as WEB19 states.

B - Incorrect: The language provides catch and throw as well as try, so this restriction is false.

C - Incorrect: WEB19 explicitly includes throw; the language does not prohibit explicitly signaling an exception.

D - Incorrect: JavaScript does support exception handling through the mechanisms described in the complete note.

Primary note proof

Study-note section WEB19

> try/catch handles thrown errors during execution; throw signals one; finally runs cleanup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 54 | lines 10-19

> 4.10.4 Handling Exception in JavaScript
> There’s a syntax construct try...catch that allows us to “catch” errors so the script
> can, instead of dying, do something more reasonable.
> The “try...catch” syntax
> The try...catch construct has two main blocks: try, and then catch:
> try {
> // code...
> } catch (err) {
> // error handling
> }

Question sources

Web Programming practice questions · Question 36

Question 84 | Q01104

Which of the following method checks if its argument is not a number?

A. isNaN()

B. nonNaN()

C. NaN()

D. None of the above

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer A: isNaN(), with coercion caveats.

WEB16 distinguishes global isNaN, which converts its argument first, from Number.isNaN, which does not. Of the offered names, isNaN is the actual check for a NaN result after coercion. It is not a universal test for whether the original argument's type is Number. Neither nonNaN nor calling NaN as a function supplies the described API.

Why each choice is right or wrong

A - Correct: Global isNaN exists and tests for NaN after numeric conversion; it is not a pure original-type test.

B - Incorrect: nonNaN() is not the check named in the supplied JavaScript API rule.

C - Incorrect: NaN is the special numeric value, not the callable checking function offered here.

D - Incorrect: isNaN is offered and matches the intended check, subject to the note's coercion caveat.

Primary note proof

Study-note section WEB-S08

> Global isNaN(value) tests whether conversion to Number produces NaN.

Study-note section WEB-S08

> NaN is itself a special numeric value, not a callable checking function.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 23-33

> 4.5.3 Data Type Conversion
> Type conversations are a very common occurrence in JavaScript. Operators and functions converts the values given to them to the right type. Mathematical operations convert value to numbers.
> - String conversion → we can use String(value) function to convert a value to string

- Numeric Conversion → happens automatically in mathematical operations and expression. We can use the Number(value) function to explicitly convert a value to a number:
let str = "123";
let num = Number(str);

Question sources

Web Programming practice questions · Question 37

Question 85 | Q01105

What does parseInt("12.8", 10) return in JavaScript?

A. Throws Error

B. It returns the decimal values in string form

C. The Number value 12

D. None of the listed option

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer C: The Number value 12

parseInt reads digits in the supplied radix until it reaches a character that cannot continue that integer representation. Here it consumes the digits before the decimal point and stops at the point, returning Number 12. It does not round the decimal value, return a string or throw merely because a decimal point appears. The concrete string avoids claiming that every decimal-looking input has a usable leading integer prefix.

Why each choice is right or wrong

A - Incorrect: A decimal point after an initial digit sequence does not require an exception.

B - Incorrect: The return value is a Number, not a string containing the decimal.

C - Correct: The radix-ten digit prefix yields Number 12.

D - Incorrect: A listed choice gives the actual result.

Primary note proof

Study-note section WEB16

parseInt("12.8",10) reads12, stopping at the dot; this is parsing, not rounding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 23-33

4.5.3 Data Type Conversion
Type conversations are a very common occurrence in JavaScript. Operators and functions converts the values given to them to the right type. Mathematical operations convert value to numbers.
- String conversion → we can use String(value) function to convert a value to string
- Numeric Conversion → happens automatically in mathematical operations and expression. We can use the Number(value) function to explicitly convert a value to a number:
let str = "123";
let num = Number(str);

Question sources

Web Programming practice questions · Question 38

Question 86 | Q01106

What is the output of below JavaScript code?alert (type of new Date());

A. Throws Error

B. object

C. Displays Nothing

D. Current Date

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer A: Syntax error: the source says type of rather than typeof.

type of new Date() is invalid JavaScript syntax; the operator is typeof, written as one word. Therefore the shown program errors before displaying an ordinary value. typeof new Date() would evaluate to "object", but that is a different expression from the code shown.

Why each choice is right or wrong

A - Correct: The spaced text type of is not the typeof operator, causing a syntax error in the literal source.

B - Incorrect: object is the result after repairing type of to typeof, which the task has not authorized.

C - Incorrect: The primary result is a syntax error, not successful execution with an intentionally empty display.

D - Incorrect: A valid typeof expression reports a type, rather than the current date value.

Primary note proof

Study-note section WEB19

A literal type of expression is not the typeof operator.

Study-note section WEB41

x=4+"4"5 is a syntax error because an operator is missing; bool ans=true and type of new Date() are syntax errors too.

Study-note section WEB16

typeof new Date() is "object"; typeof null is historically "object".

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 2-5

- Objects → are used to store a key/value pair property. Objects are created using curly braces {}
- Arrays → are regular objects and used to store indexed values. Arrays are created by using square brackets []

Web Programming module | PDF page 47 | lines 3-9

3. The write() method
- The write() method writes HTML expressions or JavaScript code to a document.
- The write() method is mostly used for testing: If it is used after an HTML document is fully loaded, it will delete all existing HTML.

Usage:
document.write(expr)

Question sources

Web Programming practice questions · Question 39

Question 87 | Q01107

What if we put ++ operator inside if condition? find the output of below code<script>var a = 10;if(a == a++)document.write(a);</script>

A. Error

B. Nothing is printed

C. 10

D. 11

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer D: 11

WEB17 traces this exact JavaScript pattern. The left operand reads 10; postfix a++ yields old 10 and then sets a to 11. Equality therefore compares 10 with 10 and is true. The body runs after the increment and writes the updated value 11. These JavaScript evaluation rules should not be replaced with C/C++ assumptions.

Why each choice is right or wrong

A - Incorrect: The expression is valid JavaScript; its specified evaluation order does not produce the supposed error.

B - Incorrect: The comparison uses old ten on both sides and is true, so the body does execute.

C - Incorrect: Ten is the compared old value, but the increment has completed before document.write reads a.

D - Correct: The equality succeeds using old values, then the body prints the updated value eleven.

Primary note proof

Study-note section WEB17

> Postfix a++ yields the old value and then increments a.

Study-note section WEB17

> If a=10, a==a++ compares10 with10, then the body sees11.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 49 | lines 9-14

> 4.7 Control Structures (Conditional and Looping Statements)
> The “if” statement:
> The if(...) statement evaluates a condition in parentheses and, if the result is true,
> executes a block of code.
> let age = 20;
> if (age > 18) alert('You are right!');

Web Programming module | PDF page 48 | lines 10-13

> - Using the let keyword → the following statement declares a variable
> let message;
> now we can put values using the assignment operator (=)
> - Using var keyword → we can also declare variable using the var keyword

Question sources

Web Programming practice questions · Question 40

Question 88 | Q01108

An inline onkeydown handler returns false for a cancelable keydown event whose default action is text insertion. Which statement correctly describes the cancellation?

A. The default text-insertion action associated with the handled keydown is canceled

B. A later keypress handler retroactively cancels the earlier keydown event

C. The keyup event is necessarily prevented from being dispatched

D. Every later keyboard event is necessarily canceled

Hint

Separate event dispatch, handler registration and cancellation of the browser default action.

Answer A: The default text-insertion action associated with the handled keydown is canceled

An appropriate inline event handler can use a false return value to cancel the handled event's default action. With the cancelable text-insertion action specified here, that default action is stopped. This does not erase an earlier event or guarantee that later keyboard events disappear. The distinction is between canceling a default behavior and suppressing an entire sequence of event dispatches; addEventListener would require preventDefault instead.

Why each choice is right or wrong

A - Correct: The inline handler convention consumes false to cancel the specified default action.

B - Incorrect: A later handler cannot undo the dispatch of an earlier event.

C - Incorrect: Canceling a default action does not guarantee suppression of keyup dispatch.

D - Incorrect: The cancellation concerns the handled default action, not every later event.

Primary note proof

Study-note section WEB21

> Returning false from an addEventListener callback does not cancel it; return false works in an appropriate inline/onsubmit property handler whose caller uses that return.

Study-note section WEB21

> For cancelable keyboard defaults, call preventDefault rather than assuming a different key event itself disappears.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 53 | lines 4-18

> 4.10.3 Events in JavaScript
> An event is a signal that something has happened. All DOM nodes generate such signals (but events are not limited to DOM).
> Here's a list of the most useful DOM events, just to take a look at:
> - click - when the mouse clicks on an element (touchscreen devices generate it

on a tap).
- contextmenu - when the mouse right-clicks on an element.
- mouseover / mouseout - when the mouse cursor comes over / leaves an element.
- mousedown / mouseup - when the mouse button is pressed / released over an element.
- mousemove - when the mouse is moved.
Event Handlers
To react on events we can assign a handler - a function that runs in case of an event.
Handlers are a way to run JavaScript code in case of user actions.

Web Programming module | PDF page 54 | lines 6-7

- passive: if true, then the handler will not call preventDefault(), we'll explain that later in Browser default actions.

Question sources

Web Programming practice questions · Question 41

Question 89 | Q01109

The _____ method of an Array object adds and/or removes elements from an array.

A. Reverse

B. Shift

C. Slice

D. Splice

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer D: splice()

WEB18 defines splice(start,deleteCount,...items) as mutating an array by removing and inserting elements. That is the combined operation in the stem. The options capitalize method names as terms; the actual JavaScript call is lowercase splice(). reverse changes order, shift removes only the first item, and slice produces a shallow copy rather than that mutation.

Why each choice is right or wrong

A - Incorrect: reverse() changes element order in place rather than adding or removing the requested entries.

B - Incorrect: shift() removes one first element; it is not the general combined insertion/removal method.

C - Incorrect: slice() returns a shallow copy of a selected range rather than mutating through insertion or removal.

D - Correct: The actual lowercase splice() method removes and optionally inserts entries in the original array.

Primary note proof

Study-note section WEB18

> slice returns a selected shallow copy; splice(start,deleteCount,...items) mutates by removal/insertion.

Study-note section WEB18

> reverse reverses in place.

Study-note section WEB18

> Array push/pop add/remove at the end; unshift/shift act at the beginning.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 51 | lines 13-17

> Methods
> - Push → appends an element to the end
> - Pop → takes an element from the end.
> - Shift → Extracts the first element of the array and returns it
> - Unshift → Add the element to the beginning of the array:

Question sources

Web Programming practice questions · Question 42

Question 90 | Q01110

The term PHP is an acronym for PHP:_____.

A. Hypertext Preprocessor

B. Hypertext multiprocessor

C. Hypertext markup Preprocessor

D. Hypertune Preprocessor

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer A: Hypertext Preprocessor

Page 60 explicitly expands PHP as Hypertext Preprocessor. No extra markup, multi-, or Hypertune term occurs in that expansion. Evidence: PDF p. 60.

Why each choice is right or wrong

A - Correct: Exact expansion.

B - Incorrect: Changes Preprocessor to multiprocessor.

C - Incorrect: Adds markup to the expansion.

D - Incorrect: Changes Hypertext to Hypertune.

Primary note proof

Study-note section WEB23

> PHP means PHP: Hypertext Preprocessor.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 4-8

> PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the

Question sources

Web Programming practice questions · Question 43

Question 91 | Q01111

PHP is a _____ language?

A. user-side scripting

B. client-side scripting

C. server-side scripting

D. Both B and C

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer C: server-side scripting

In the web execution model used here, PHP runs on the server, generates responses and can access databases. The browser receives its output rather than executing the PHP source as its client-side script. Client-side interactivity is supplied by languages such as JavaScript. Thus server-side scripting is the matching category, and the combined alternative incorrectly adds client-side execution.

Why each choice is right or wrong

A - Incorrect: User-side is not the server-execution category described for PHP.

B - Incorrect: Client-side scripting runs in the browser, whereas the described PHP processing runs on the server.

C - Correct: PHP is the server-side scripting language in this web application model.

D - Incorrect: This combines the correct server role with a client-side role that does not apply to PHP source in the described model.

Primary note proof

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Study-note section WEB23

> It can emit ordinary text and HTML.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 4-10

> PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in

scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Web Programming practice questions · Question 44

Question 92 | Q01112

PHP files have a default file extension of_____.

A. .html

B. .xml

C. .php

D. .hphp

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer C: .php

The module explicitly states that PHP documents end with .php by default and are passed to the PHP processor. Evidence: PDF p. 60.

Why each choice is right or wrong

A - Incorrect: HTML document extension, not the stated PHP default.

B - Incorrect: Not the module PHP extension.

C - Correct: Exact default extension.

D - Incorrect: Not the module PHP extension.

Primary note proof

Study-note section WEB23

> Use <?php ...

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 20-22

> By default, PHP documents end with the extension .php. When a web server encounters this extension in a requested file, it automatically passes it to the PHP processor. At its very simplest, a PHP document will output only HTML.

Question sources

Web Programming practice questions · Question 45

Question 93 | Q01113

Which of the following is the correct syntax of php?

A. <?php >

B. <php >

C. ?php ?

D. <?php ?>

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer D: <?php ?>

The standard PHP delimiter pair shown on p.60 is <?php followed by ?>. Both markers are necessary to match the demonstrated embedded-code form. Evidence: PDF p. 60.

Why each choice is right or wrong

A - Incorrect: Closing marker lacks question mark.

B - Incorrect: Opening and closing markers are incomplete.

C - Incorrect: Neither marker has angle brackets.

D - Correct: Matches both standard markers.

Primary note proof

Study-note section WEB23

> Use <?php ...

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 23-27

> 5.2 Basic PHP Syntax
> The most universally effective PHP syntax is:
> <?php
> //php script
> ?>

Question sources

Web Programming practice questions · Question 46

Question 94 | Q01114

Which of the below statements is equivalent to $sub -= $sub?

A. $sub = $sub

B. $sub = $sub -$sub

C. $sub = $sub - 1

D. $sub = $sub - $sub - 1

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer B: $sub = $sub - $sub

WEB26 explicitly expands $sub-=$sub into $sub=$sub-$sub. Compound subtraction uses the existing left-side value minus the stated right-side value and stores the result back. Here both references name the same value; no constant one is subtracted. The expression is therefore not decrement and does not preserve the old value by simple reassignment.

Why each choice is right or wrong

A - Incorrect: Simple self-assignment leaves the value unchanged rather than subtracting the right-hand operand.

B - Correct: This is the exact expansion of compound subtraction with $sub on both sides.

C - Incorrect: Subtracting one is decrement; the actual right-hand operand is $sub rather than the constant one.

D - Incorrect: The extra subtraction of one is absent from the compound assignment and changes its meaning.

Primary note proof

Study-note section WEB26

> $sub-=$sub is $sub=$sub-$sub, not decrement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 64 | lines 10-12

> ➢ The value of a variable is the value of its most recent assignment.
> ➢ Variables are assigned with the = operator, with the variable on the left-hand side and the expression to be evaluated on the right.

Web Programming module | PDF page 65 | lines 8-12

> Arithmetic Operators:
> Arithmetic operators do what you would expect. They are used to perform mathematics. You can use them for the main four operations (plus, minus, multiply, and

divide) as well as to find a modulus (the remainder after a division) and to increment or decrement a value.

Question sources

Web Programming practice questions · Question 47

Question 95 | Q01115

Which PHP statement reliably outputs the literal characters $lfc, regardless of the variable's current value?

A. echo "$lfc";

B. echo "$$lfc";

C. echo '$lfc';

D. echo "$lfc;";

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: echo '$lfc';

Double-quoted PHP strings interpolate variable names, so the unescaped occurrences of $lfc depend on the variable's value. A single-quoted string normally preserves the dollar name literally and therefore emits the requested characters. Extra dollar signs or quoted semicolons do not turn the remaining double-quoted choices into reliable literal output. The option changes the quoting rather than assuming a convenient variable value.

Why each choice is right or wrong

A - Incorrect: Double quotes interpolate the variable's value.

B - Incorrect: The extra dollar sign does not prevent interpolation of $lfc.

C - Correct: Single quotes preserve the dollar name as literal text.

D - Incorrect: The variable is interpolated and the semicolon inside the string is also output.

Primary note proof

Study-note section WEB24

Double-quoted strings interpolate variables and escapes such as \n; single quotes normally preserve dollar names literally.

Study-note section WEB24

To emit literal $lfc use echo '$lfc'; or escape the dollar in double quotes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 62 | lines 27-33

In PHP user defined variable can be declared as:
$var_name = value;
Example
<?php
// Declaring variables
$txt = "Hello World!";
$number = 10;

Web Programming module | PDF page 63 | lines 5-18

5.3 PHP Output Statements
The two most basic constructs for printing to output are echo and print. Their language status is somewhat confusing, because they are basic constructs of the PHP language, rather than being functions. As a result, they can be used either with parentheses or without them.
Echo
The simplest use of echo is to print a string as argument.
For example:
echo “This will print in the user’s browser window.”;
Or equivalently:
echo(“This will print in the user’s browser window.”);
You can also give multiple arguments to the unparenthesized version of echo, separated by commas, as in:
echo “This will print in the “, “user‟s browser window.”;

Question sources

Web Programming practice questions · Question 48

Question 96 | Q01116

Which of the below symbol is a newline character?

A. \r

B. \n

C. /r

D. /n

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: \n

WEB24 identifies \n among escapes interpreted in PHP double-quoted strings. That sequence represents a newline; a forward slash does not introduce the escape. \r denotes a different line-control character rather than the newline sequence requested. Interpretation depends on string context, since single-quoted strings generally preserve such escapes rather than processing them the same way.

Why each choice is right or wrong

A - Incorrect: \r is the carriage-return escape, a different line-control character from the newline sequence requested.

B - Correct: \n is the newline escape sequence recognized in the note's double-quoted string context.

C - Incorrect: A forward slash does not introduce the newline escape, so /r is ordinary text here.

D - Incorrect: The newline escape uses a backslash; /n with a forward slash is ordinary text.

Primary note proof

Study-note section WEB-S06

> In double-quoted PHP strings, backslash-n represents a line feed and backslash-r represents a carriage return.

Study-note section WEB-S06

> The written sequences are \n and \r; forward-slash sequences such as /n are ordinary text.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 15 | lines 33-34

> • line break - inserts a new line.
> Format: <br>. E.g., Line one <br> line two <br> line three <br> line four

Web Programming module | PDF page 63 | lines 11-18

> The simplest use of echo is to print a string as argument.
> For example:
> echo “This will print in the user’s browser window.”;
> Or equivalently:
> echo(“This will print in the user’s browser window.”);
> You can also give multiple arguments to the unparenthesized version of echo,
> separated by commas, as in:
> echo “This will print in the “, “user‟s browser window.”;

Question sources

Web Programming practice questions · Question 49

Question 97 | Q01117

PHP files can contain _____ code.

A. text

B. HTML

C. python

D. Both A and B

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer D: Both A and B

PHP can be embedded within ordinary HTML (p.60); the output examples contain text strings (p.63). Thus text and HTML can coexist in a PHP document. Evidence: PDF p. 60, PDF p. 63.

Why each choice is right or wrong

A - Incorrect: Text can occur, but A alone omits HTML.

B - Incorrect: HTML can occur, but B alone omits text.

C - Incorrect: The module does not define Python embedding or execution.

D - Correct: Combines the supported text and HTML forms.

Primary note proof

Study-note section WEB23

> It can emit ordinary text and HTML.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 28-32

> If you use this style, you can be positive that your tags will always be correctly interpreted. The way you use this tag is quite flexible. Some programmers enclose everything between this opening and closing tag. Others, however, choose to insert only wherever dynamic scripting is required, leaving the rest of the document in standard HTML.

Web Programming module | PDF page 63 | lines 11-18

> The simplest use of echo is to print a string as argument.
> For example:
> echo “This will print in the user’s browser window.”;
> Or equivalently:
> echo(“This will print in the user’s browser window.”);
> You can also give multiple arguments to the unparenthesized version of echo,

separated by commas, as in:
echo “This will print in the “, “user‟s browser window.”;

Question sources

Web Programming practice questions · Question 50

Question 98 | Q01118

What will be the output of the following PHP code?
<?php
$x = 8;
$y = 8.0;
echo ($x === $y);
?>

A. 8 === 8

B. No Output

C. 1

D. 0

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer B: No output characters.

In PHP, 8===8.0 is false because int and float differ, despite equal numeric values. echo false emits an empty string. Consequently the shown code produces no output characters, rather than printing zero. The comparison expression itself is evaluated and is not emitted as source text.

Why each choice is right or wrong

A - Incorrect: echo evaluates the comparison instead of printing its source expression.

B - Correct: Strict identity fails for int versus float, and echo false emits no characters.

C - Incorrect: One is the echo representation of true; the strict identity comparison here is false.

D - Incorrect: Although the comparison is false, echo false emits an empty string rather than the digit zero.

Primary note proof

Study-note section WEB24

PHP 8==8.0 is true, but 8===8.0 is false because types differ; echo false emits an empty string, echo true emits1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 63 | lines 27-33

> 5.4 Data Types and Variables in PHP
> PHP is a very loosely typed language. This means that variables do not have to be declared before they are used, and that PHP always converts variables to the type required by their context when they are accessed. Variables can store data of different types, and different data types can do different things. PHP supports the following data types: String, Integer, Float (floating point numbers - also called double), Boolean, Array, Object, NULL and Resource

Web Programming module | PDF page 65 | lines 13-16

> Logical Operators:
> As a rule, if something has a TRUE or FALSE value, it can be input to a logical operator. A logical operator takes two true-or-false inputs and produces a true-or-false result. The PHP logical operators are sued to combine conditional statements.

Question sources

Web Programming practice questions · Question 51

Question 99 | Q01119

The script that executes at the browser side is called _____

A. Client-side scripting

B. Server-side scripting

C. Both A and B

D. None of the above

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Client-side scripting

The module distinguishes client/browser behavior from server-side PHP. Code executing in the browser is client-side scripting in this model. Evidence: PDF p. 9, PDF p. 60, PDF p. 56.

Why each choice is right or wrong

A - Correct: Browser-side matches client-side.

B - Incorrect: PHP server processing happens on the server.

C - Incorrect: The stated execution site is the client, not both.

D - Incorrect: A matches the documented client/server distinction.

Primary note proof

Study-note section WEB01

> A browser/user agent requests resources and renders HTML/CSS, then runs permitted JavaScript.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 9 | lines 2-4

> • Web pages are files stored on computers called Web servers.
> • Computers reading the Web pages are called Web clients.
> • Web clients view the pages with a program called a Web browser.

Web Programming module | PDF page 60 | lines 11-16

> Client-side scripting is the glamorous, eye-catching part of Web development. In contrast, server-side scripting is invisible to the user. Server-side Web scripting is mostly about connecting Web sites to back-end servers, such as databases. This enables the following types of two-way communication:
> ➢ Server to client: Web pages can be assembled from back end-server output.
> ➢ Client to server: Customer-entered information can be acted upon.

Web Programming module | PDF page 56 | lines 7-9

> The Browser Object Model (BOM) is a browser-specific convention referring to all the objects exposed by the web browser. The BOM allows JavaScript to “interact with” the browser.

Question sources

Web Programming practice questions · Question 52

Question 100 | Q01120

How many basic PHP output language constructs are counted when callable output functions such as printf and print_r are excluded?

A. 1

B. 2

C. 3

D. 4

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: 2

The note identifies echo and print as two basic output constructs. This count is about that language-construct category, rather than every way a PHP program can produce output. Functions such as printf and print_r provide additional output mechanisms but do not increase the specified count of constructs. The revised category therefore supports one numeric choice without implying that PHP has only two output mechanisms altogether.

Why each choice is right or wrong

A - Incorrect: Both echo and print belong to the specified category, so one is too few.

B - Correct: The category contains the two named basic constructs.

C - Incorrect: This would count an additional mechanism outside the specified construct category.

D - Incorrect: This would mix other output mechanisms into the restricted category.

Primary note proof

Study-note section WEB24

> They are two basic output constructs, not the only output mechanisms; printf, print_r and others exist.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 63 | lines 5-9

> 5.3 PHP Output Statements
> The two most basic constructs for printing to output are echo and print. Their language status is somewhat confusing, because they are basic constructs of the PHP language, rather than being functions. As a result, they can be used either with parentheses or without them.

Web Programming module | PDF page 69 | lines 34-35

> We have also used the print_r function (which prints out the contents of a variable, array, or object) is used to verify that the array has been correctly populated.

Question sources

Web Programming practice questions · Question 53

Web Programming practice questions · Question 54

Question 101 | Q01122

How many return value does echo statement has?

A. 1

B. 2

C. 3

D. 0

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer D: 0

Page 63 contrasts print, which returns a value, with echo, which does not. Thus echo has zero return values; zero here counts return values rather than describing a returned numeric zero. Evidence: PDF p. 63.

Why each choice is right or wrong

A - Incorrect: Would give echo the return behavior assigned to print.

B - Incorrect: No two return values are described.

C - Incorrect: No three return values are described.

D - Correct: No return value, expressed as count zero.

Primary note proof

Study-note section WEB24

> echo is a language construct with no return value and can emit comma-separated expressions.

Study-note section WEB24

> print takes one expression and returns1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 63 | lines 21-26

> Print
> The command print is very similar to echo, with two important differences:
> ✦ unlike echo, print can accept only one argument.
> ✦ unlike echo, print returns a value, which represents whether the print statement succeeded. The value returned by print will be 1 if the printing was successful and 0 if unsuccessful.

Question sources

Web Programming practice questions · Question 55

Question 102 | Q01123

What exact raw response text does this PHP emit, before any browser HTML rendering?
<?php
echo "Hello world </br> I am learning PHP at letsfindcourse";
?>

A. Hello world

B. Hello world </br> I am learning PHP at letsfindcourse

C. Hello worldI am learning PHP at letsfindcourse

D. Error

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer B: Hello world </br> I am learning PHP at letsfindcourse

echo emits the characters of its string without automatically adding or removing spaces. Because the question asks for raw response text, the break-like markup remains present exactly as written; browser error recovery is irrelevant to this output. Both phrases and the spaces around the markup are preserved. The HTML fragment's unusual closing form does not make the surrounding PHP string syntactically invalid.

Why each choice is right or wrong

A - Incorrect: This omits the markup and the second phrase.

B - Correct: This preserves every character in the emitted string.

C - Incorrect: This omits the intervening spaces, markup and separation.

D - Incorrect: The markup is inside a valid PHP string and causes no PHP parse error.

Primary note proof

Study-note section WEB24

> PHP sends markup as response characters; a browser then interprets it.

Study-note section WEB24

> Neither automatically inserts spaces/newlines.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 63 | lines 11-15

> The simplest use of echo is to print a string as argument.
> For example:

echo “This will print in the user’s browser window.”;
Or equivalently:
echo(“This will print in the user’s browser window.”);

Web Programming module | PDF page 15 | lines 33-34

• line break - inserts a new line.
Format: <br>. E.g., Line one <br> line two <br> line three <br> line four

Question sources

Web Programming practice questions · Question 56

Question 103 | Q01124

What will be the output of the following PHP code?
<?php
$five = 5;
print($five);
print $five;
?>

A. 05

B. 55

C. 50

D. Error

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer B: 55

print accepts one argument and may be written with or without parentheses (p.63). Both statements print the assigned 5, with no intervening separator, giving 55. Evidence: PDF p. 63, PDF p. 63, PDF p. 64.

Why each choice is right or wrong

A - Incorrect: First output is 5, not zero.

B - Correct: Two successive prints of 5 concatenate in output.

C - Incorrect: Second output is also 5.

D - Incorrect: Both print forms are explicitly permitted.

Primary note proof

Study-note section WEB24

Two prints of5 emit55; "I"."am" emitsIam.

Study-note section WEB-S06

The basic print and echo constructs can be written with or without parentheses when their syntax permits it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 63 | lines 5-9

5.3 PHP Output Statements
The two most basic constructs for printing to output are echo and print. Their language status is somewhat confusing, because they are basic constructs of the PHP language, rather than being functions. As a result, they can be used either with parentheses or without them.

Web Programming module | PDF page 63 | lines 21-26

Print
The command print is very similar to echo, with two important differences:
✦ unlike echo, print can accept only one argument.
✦ unlike echo, print returns a value, which represents whether the print statement succeeded. The value returned by print will be 1 if the printing was successful and 0 if unsuccessful.

Web Programming module | PDF page 64 | lines 10-12

➢ The value of a variable is the value of its most recent assignment.
➢ Variables are assigned with the = operator, with the variable on the left-hand side and the expression to be evaluated on the right.

Question sources

Web Programming practice questions · Question 57

Question 104 | Q01125

What will be the output of the following PHP code?
<?php
print("I"."am"."learning"."php");
?>

A. Iamlearningphp

B. I am learning php

C. Nothing

D. error

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer A: Iamlearningphp

The PHP examples join strings and variables using dot (p.72). Joining I, am, learning and php inserts no spaces, so the resulting single argument is Iamlearningphp and print emits it. Evidence: PDF p. 72, PDF p. 63.

Why each choice is right or wrong

A - Correct: Preserves all characters with no added spaces.

B - Incorrect: No operand contains a space.

C - Incorrect: print produces output, not silence.

D - Incorrect: Single joined argument fits print.

Primary note proof

Study-note section WEB24

> Dot concatenates; + is arithmetic.

Study-note section WEB24

> Neither automatically inserts spaces/newlines.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 72 | lines 19-22

> function info($name,$age,$salary){
> echo “name:”.$name.”
”;
> echo “age:”.$age.”
”;
> echo “salary:”.$salary.”
”; }

Web Programming module | PDF page 63 | lines 21-26

> Print
> The command print is very similar to echo, with two important differences:
> ✦ unlike echo, print can accept only one argument.
> ✦ unlike echo, print returns a value, which represents whether the print statement succeeded. The value returned by print will be 1 if the printing was successful and 0 if unsuccessful.

Question sources

Web Programming practice questions · Question 58

Question 105 | Q01126

which of the following is correct echo statement?

A. echo "<h2>PHP is Fun!</h2>";

B. echo "PHP is Fun!
";

C. echo "PHP ", "is ", "Fun!";

D. All of the above

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer D: All of the above

Echo accepts quoted strings, including HTML in generated output, and multiple comma-separated arguments when unparenthesized (p.63). All three snippets meet those forms. Evidence: PDF p. 63, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: Valid string containing heading markup.

B - Incorrect: Valid string containing a line-break tag.

C - Incorrect: Valid unparenthesized multiple-argument echo.

D - Correct: All A-C are valid under the cited echo rules.

Primary note proof

Study-note section WEB24

> echo is a language construct with no return value and can emit comma-separated expressions.

Study-note section WEB24

> PHP sends markup as response characters; a browser then interprets it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 63 | lines 11-20

> The simplest use of echo is to print a string as argument.
> For example:
> echo “This will print in the user’s browser window.”;
> Or equivalently:
> echo(“This will print in the user’s browser window.”);
> You can also give multiple arguments to the unparenthesized version of echo,
> separated by commas, as in:
> echo “This will print in the “, “user‟s browser window.”;

The parenthesized version, however, will not accept multiple arguments:
echo (“This will produce a “, “PARSE ERROR!”);

Web Programming module | PDF page 60 | lines 20-22

By default, PHP documents end with the extension .php. When a web server encounters this extension in a requested file, it automatically passes it to the PHP processor. At its very simplest, a PHP document will output only HTML.

Question sources

Web Programming practice questions · Question 59

Question 106 | Q01127

What will be the output of the following PHP code?
<?php
echo "This is Letsfindcourse";
?>

A. This is Letsfindcourse

B. This is Letsfindcourse(Letsfindcourse in italic)

C. This is

D. Error

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer B: This is Letsfindcourse(Letsfindcourse in italic)

Echo emits the quoted HTML string (p.63); the i element renders its content in italics (p.15). Consequently the full text is displayed with Letsfindcourse italicized. Evidence: PDF p. 63, PDF p. 15.

Why each choice is right or wrong

A - Incorrect: Text is present but this omits the requested formatting distinction.

B - Correct: States both the text and italic formatting.

C - Incorrect: Incorrectly drops the i element content.

D - Incorrect: The quoted string and i tags fit the documented forms.

Primary note proof

Study-note section WEB24

PHP sends markup as response characters; a browser then interprets it.

Study-note section WEB-S07

In the browser’s default presentation, an i element typically displays its text in italics.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 63 | lines 11-15

The simplest use of echo is to print a string as argument.
For example:
echo “This will print in the user’s browser window.”;
Or equivalently:
echo(“This will print in the user’s browser window.”);

Web Programming module | PDF page 15 | lines 22-23

Italics - makes a text appear in italics
Format: ... or .... E.g., a text in italics

Question sources

Web Programming practice questions · Question 60

Question 107 | Q01128

Any variables declared in PHP must begin with a _____?

A. .

B. #

C. &

D. $

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer D: $

All PHP variables have a leading dollar sign according to p.64. Evidence: PDF p. 64.

Why each choice is right or wrong

A - Incorrect: Dot is not the variable prefix.

B - Incorrect: Hash is not the variable prefix.

C - Incorrect: Ampersand is not the variable prefix.

D - Correct: Exact mandatory leading symbol.

Primary note proof

Study-note section WEB23

> Variables begin with $, then a letter/underscore and subsequent letters/digits/underscores; $AGE18 works but $18AGE does not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 64 | lines 6-9

> All variables in PHP are denoted with a leading dollar sign ($).
> All variables in PHP start with a leading $ sign. After the initial $, variable names must be composed of letters (uppercase or lowercase), digits (0-9), and underscore characters (_). Furthermore, the first character after the $ may not be a number.

Question sources

Web Programming practice questions · Question 61

Question 108 | Q01129

A variable can have _____?

A. long descriptive names

B. short names

C. Both A and B

D. None of the above

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer C: Both A and B

After $, identifiers may contain letters, digits and underscores, with no stated fixed short length; both short and descriptive names fit this rule. Names must still satisfy the first-character restriction. Evidence: PDF p. 64.

Why each choice is right or wrong

A - Incorrect: Allowed but not the only allowed length style.

B - Incorrect: Allowed but not the only allowed length style.

C - Correct: Both comply when characters are valid.

D - Incorrect: The naming rule permits both examples of name length.

Primary note proof

Study-note section WEB23

> Variables begin with $, then a letter/underscore and subsequent letters/digits/underscores; $AGE18 works but $18AGE does not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 64 | lines 6-9

> All variables in PHP are denoted with a leading dollar sign ($).
> All variables in PHP start with a leading $ sign. After the initial $, variable names must be composed of letters (uppercase or lowercase), digits (0-9), and underscore characters (_). Furthermore, the first character after the $ may not be a number.

Question sources

Web Programming practice questions · Question 62

Question 109 | Q01131

Which characters may start a PHP variable identifier immediately after its mandatory $ sign? Select all that apply.

A. letter

B. underscore

C. A digit

D. A space

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer A, B: letter; underscore

A PHP variable begins with a dollar sign followed by a letter or underscore. Digits may occur later in the identifier but may not be the first character after the sigil. A space cannot form that initial identifier character either. The question specifies the position being tested and replaces the vague no-numbers and aggregate wording with concrete candidate characters, allowing both valid initial-character classes to be selected.

Why each choice is right or wrong

A - Correct: A letter is allowed immediately after the dollar sign.

B - Correct: An underscore is also allowed immediately after the dollar sign.

C - Incorrect: A digit may appear later, but cannot start the identifier.

D - Incorrect: A space is not an allowed initial identifier character.

Primary note proof

Study-note section WEB23

> Variables begin with $, then a letter/underscore and subsequent letters/digits/underscores; $AGE18 works but $18AGE does not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 64 | lines 6-9

> All variables in PHP are denoted with a leading dollar sign ($).
> All variables in PHP start with a leading $ sign. After the initial $, variable names must be composed of letters (uppercase or lowercase), digits (0-9), and underscore characters (_). Furthermore, the first character after the $ may not be a number.

Question sources

Web Programming practice questions · Question 64

Question 110 | Q01132

PHP variables are case-sensitive?

A. True

B. False

C. Only the variable named sum is case-sensitive

D. None of the above

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer A: True

PHP variable names are case-sensitive generally. Two names that differ in letter case can identify different variables, regardless of whether the name is sum or something else. True therefore gives the general rule. False contradicts it, and the sum-only restriction is false because case sensitivity is not limited to that name. None is incorrect because the true general statement is offered.

Why each choice is right or wrong

A - Correct: PHP variable names are case-sensitive as a general rule, exactly as WEB23 states.

B - Incorrect: False contradicts the explicit case-sensitivity rule for PHP variable names.

C - Incorrect: Case sensitivity applies to PHP variable names generally, not only to sum.

D - Incorrect: True is offered and matches the note, so None is not applicable.

Primary note proof

Study-note section WEB23

> Names are case-sensitive.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 64 | lines 6-9

> All variables in PHP are denoted with a leading dollar sign ($).
> All variables in PHP start with a leading $ sign. After the initial $, variable names must be composed of letters (uppercase or lowercase), digits (0-9), and underscore characters (_). Furthermore, the first character after the $ may not be a number.

Question sources

Web Programming practice questions · Question 65

Question 111 | Q01133

In the common PHP teaching taxonomy that counts persistent function-local storage as an additional category alongside visibility categories, how many variable categories are counted?

A. 2

B. 3

C. 1

D. 4

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: 3

The stated teaching taxonomy counts local, global and static as three categories. Global and local distinguish visibility, while a static local describes persistence across calls within the execution lifecycle. The question makes this mixed teaching convention explicit rather than treating static storage as an independent visibility scope. Counting only visibility would answer a different question, which is why the narrower numeric choice is now justified.

Why each choice is right or wrong

A - Incorrect: Two counts local/global visibility alone and omits the stated static category.

B - Correct: The specified teaching taxonomy counts local, global and static.

C - Incorrect: One omits the distinctions explicitly included in the taxonomy.

D - Incorrect: No fourth category is included in the stated convention.

Primary note proof

Study-note section WEB25

> Common “three scopes” teaching counts local, global and static, but state the taxonomy.

Study-note section WEB25

> A static local retains its value across calls during the execution lifecycle; static storage is not simply another visibility scope.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 62 | lines 12-18

> PHP provides a large number of predefined variables to any script which it runs.
> PHP provides an additional set of predefined arrays containing variables from the web server the environment, and user input. These new arrays are called superglobals –
> All the following variables are automatically available in every scope.
> - $GLOBALS → Contains a reference to every variable which is currently

available within the global scope of the script. The keys of this array are the names of the global variables.

Question sources

Web Programming practice questions · Question 66

Question 112 | Q01134

choose the incorrect variable name in php?

A. $_AGE

B. $18AGE

C. $AGE18

D. $_AGE18

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B: $18AGE

PHP variable names may include underscores, letters and digits, but the first character after $ cannot be numeric (p.64). Only $18AGE violates that rule. Evidence: PDF p. 64.

Why each choice is right or wrong

A - Incorrect: Underscore then letters is permitted.

B - Correct: Starts with digit 1 after $, expressly forbidden.

C - Incorrect: Letter start and later digits are permitted.

D - Incorrect: Underscore start and later digits are permitted.

Primary note proof

Study-note section WEB23

> Variables begin with $, then a letter/underscore and subsequent letters/digits/underscores; $AGE18 works but $18AGE does not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 64 | lines 6-9

> All variables in PHP are denoted with a leading dollar sign ($).
> All variables in PHP start with a leading $ sign. After the initial $, variable names must be composed of letters (uppercase or lowercase), digits (0-9), and underscore characters (_). Furthermore, the first character after the $ may not be a number.

Question sources

Web Programming practice questions · Question 67

Question 113 | Q01135

A variable declared _____ has a GLOBAL SCOPE?

A. outside program

B. inside function

C. outside function

D. None of the Above

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: Outside a function

WEB25 directly states that a variable outside a function has global scope. A function's own locals and by-value parameters are separate bindings; functions must explicitly refer to globals through global or $GLOBALS when needed. Outside program is not a declaration location, and inside function describes the ordinary local-binding setting rather than the requested global one.

Why each choice is right or wrong

A - Incorrect: Outside program is not a meaningful place for a declaration within the executing PHP source.

B - Incorrect: Ordinary variables declared inside a function are local to that function, not automatically global.

C - Correct: A declaration outside a function creates a global binding under the rule given in WEB25.

D - Incorrect: Outside function is explicitly offered and matches the global-scope rule.

Primary note proof

Study-note section WEB25

> A variable outside a function has global scope.

Study-note section WEB25

> A function's locals and by-value parameters are separate bindings; use global $x or $GLOBALS["x"] to refer to a global explicitly.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 62 | lines 15-18

> All the following variables are automatically available in every scope.
> - $GLOBALS → Contains a reference to every variable which is currently available within the global scope of the script. The keys of this array are the names of the global variables.

Question sources

Web Programming practice questions · Question 68

Question 114 | Q01136

Which of the following will be output of the following code?

```
<?php
function myTest() {
static $x;
echo $x;
}
myTest();
?>
```

A. 0

B. 1

C. No Output

D. Error

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer C: No output

WEB25 states that an uninitialized static local begins as null and echo emits no characters for it. The function declares static $x but never assigns or increments it, then echoes that null value on its only call. The source is valid, so neither an error nor an assumed numeric initialization is the supported result.

Why each choice is right or wrong

A - Incorrect: The uninitialized static local begins as null; echo does not print an assumed numeric zero.

B - Incorrect: No assignment or increment produces one anywhere in the function.

C - Correct: Echoing the initial null static local emits no output characters.

D - Incorrect: Declaring an uninitialized static local is valid PHP and does not itself produce an error.

Primary note proof

Study-note section WEB25

> An uninitialized static local begins as null, so echo emits no characters.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 64 | lines 13-17

> ➢ Variables can, but do not need, to be declared before assignment.
> ➢ Variables have no intrinsic type other than the type of their current value.
> In PHP, because types are associated with values rather than variables, no such declaration is necessary-the first step in using a variable is to assign it a value.
> ✓ Variables used before they are assigned have default values.

Question sources

Web Programming practice questions · Question 70

Question 115 | Q01137

Which of the following is NOT a super global variable ?

A. $_SERVER

B. $_GLOBAL

C. $_GET

D. $_POST

Hint

Identify the data source or security responsibility precisely, and check whether a combined source or a different control could also qualify.

Answer B: $_GLOBAL

The predefined list includes $_SERVER, $_GET and $_POST, while the global map is $GLOBALS, not $_GLOBAL. B confuses its spelling and is not the listed superglobal. Evidence: PDF p. 62.

Why each choice is right or wrong

A - Incorrect: Explicitly listed.

B - Correct: Wrong spelling for $GLOBALS; not the module superglobal.

C - Incorrect: Explicitly listed.

D - Incorrect: Explicitly listed.

Primary note proof

Study-note section WEB27

> $GLOBALS is the actual global-symbol array; $_GLOBAL is not the built-in name.

Study-note section WEB27

> $_GET contains query parameters; $_POST contains supported form POST data; $_FILES upload metadata; $_SERVER request/environment details; $_COOKIE cookie values; $_SESSION session data after startup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 62 | lines 12-26

PHP provides a large number of predefined variables to any script which it runs.
PHP provides an additional set of predefined arrays containing variables from the web server the environment, and user input. These new arrays are called superglobals –
All the following variables are automatically available in every scope.
- $GLOBALS → Contains a reference to every variable which is currently available within the global scope of the script. The keys of this array are the names of the global variables.
- $_SERVER → This is an array containing information such as headers, paths, and script locations. The entries in this array are created by the web server. There is no guarantee that every web server will provide any of these. See next section for a complete list of all the SERVER variables.
- $_GET → An associative array of variables passed to the current script via the HTTP GET method.
- $_POST → An associative array of variables passed to the current script via the HTTP POST method.

Question sources

Web Programming practice questions · Question 71

Question 116 | Q01138

Predict the output of the following code snippet :

```
<?php
$i=10;
function sample($i)
{
$i++;
echo $i.",";
}
echo $i.",";
GLOBAL $i;
sample($i);
echo $i;
?>
```

A. 11,11,11

B. 10,11,11

C. 10,11,10

D. 10,10,10

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer C: 10,11,10

WEB25 traces this exact pattern: the initial global print emits 10, the function increments its local by-value parameter to 11, and the caller remains 10 afterward. The source writes commas after the first two values. A top-level GLOBAL statement does not turn the function's parameter into a reference, so the final output is 10,11,10.

Why each choice is right or wrong

A - Incorrect: The first print occurs before any increment and therefore emits ten rather than eleven.

B - Incorrect: The function increments a by-value local parameter, so the final caller value remains ten.

C - Correct: The caller prints ten, the local parameter prints eleven, and the unchanged caller prints ten.

D - Incorrect: The function does increment its local parameter, so the middle output is eleven rather than ten.

Primary note proof

Study-note section WEB25

> If global $i=10 and sample($i) increments/prints its by-value parameter, the caller's $i stays10.

Study-note section WEB25

> A global statement outside the function does not turn its value parameter into a reference.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 72 | lines 7-22

> Function definitions have the following form:
> function function_name([parameter [, ...]])
> {
> // Statements
> }
> That is, function definitions have four parts:
> - A definition starts with the word function.
> - A name follows, which must start with a letter or underscore, followed by
> any number of letters, numbers, or underscores.
> - The parentheses are required.
> - One or more parameters, separated by commas, are optional
> Example:
> function info($name,$age,$salary){

```
echo “name:”.$name.”<br>”;
echo “age:”.$age.”<br>”;
echo “salary:”.$salary.”<br>”; }
```

Web Programming module | PDF page 62 | lines 15-18

All the following variables are automatically available in every scope.
- $GLOBALS → Contains a reference to every variable which is currently available within the global scope of the script. The keys of this array are the names of the global variables.

Web Programming module | PDF page 65 | lines 8-12

Arithmetic Operators:
Arithmetic operators do what you would expect. They are used to perform mathematics. You can use them for the main four operations (plus, minus, multiply, and divide) as well as to find a modulus (the remainder after a division) and to increment or decrement a value.

Question sources

Web Programming practice questions · Question 72

Question 117 | Q01139

Can we change the value of constant during the execution of the script.

A. can

B. can not

C. depends on data type

D. depends on variable

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B: Cannot redefine the constant during ordinary script execution.

WEB23 says constants cannot normally be redefined during execution. That immutability is the property being tested, rather than whether a variable can receive another value. The offered data-type and variable-dependent alternatives do not override the constant binding rule. Therefore can not is the appropriate general answer in the note’s PHP context.

Why each choice is right or wrong

A - Incorrect: Ordinary constant bindings cannot be reassigned as variables can during script execution.

B - Correct: The note explicitly says constants cannot normally be redefined during execution.

C - Incorrect: The redefinition restriction is a binding rule rather than a choice determined by the constant's data type.

D - Incorrect: A constant's ordinary redefinition rule does not depend on the value of some variable.

Primary note proof

Study-note section WEB23

> Constants cannot normally be redefined during execution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 19-22

> - Using const keyword → to declare a constant variable we can use const
> instead of var and let. Variables declared with const can not be changed
> once they are declared.
> const PI = 3.14;

Web Programming module | PDF page 72 | lines 2-5

> print("hello world")
> The parenthesis tells php that you are referring to a function. Otherwise, PHP
> thinks you are referencing to a constant. Functions can take any number of arguments,
> including zero

Question sources

Web Programming practice questions · Question 73

Question 118 | Q01167

The element is used to contain characters that should be superscript.

A. a | <sup> v

B. i? <sub> x

C. /@| <ins> x

D. | None of the above x

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer A: a | <sup> v

Page 16 pairs sub with subscript and sup with superscript. Ignoring the source annotation debris around options, A contains the required sup tag. Evidence: PDF p. 16.

Why each choice is right or wrong

A - Correct: Contains the documented superscript tag.

B - Incorrect: sub is subscript.

C - Incorrect: ins is not the documented superscript form.

D - Incorrect: A provides the required element.

Primary note proof

Study-note section WEB06

> <sub>/<sup> mark sub/superscripts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 16 | lines 9-12

> sub/sup - define either a subscript or a superscript
> Format: _{...} ; ^{...}
> E.g.
> X₁² + 2X₃

Question sources

Computer Science Exit Exam practice questions · Question 28

Question 119 | Q01168

You can add a horizontal rule between sections using tag of

A. a |
 x

B. i? <hr> v

C. '@| All of the above ><

D. | None of the above x

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer B: i? <hr> v

The horizontal-rule section explicitly gives <hr>. A br instead inserts a line break, so the all-of-above alternative fails. Evidence: PDF p. 16, PDF p. 15.

Why each choice is right or wrong

A - Incorrect: br is a text line break.

B - Correct: hr is the horizontal rule.

C - Incorrect: Not all: br is a different operation.

D - Incorrect: B supplies the documented tag.

Primary note proof

Study-note section WEB06

> <p> marks paragraphs;
 is a line break and <hr> a thematic break.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 16 | lines 2-8

> • horizontal rule - inserts a horizontal line
> Format: <hr>
> Attributes:
> ✓ width=“width” {absolute: in pixels or relative: in %}
> ✓ noshade
> ✓ color=“color” {browser dependent}
> E.g. <hr width=“75%” noshade color=“#FF0000”>

Web Programming module | PDF page 15 | lines 33-34

> • line break - inserts a new line.
> Format: <br>. E.g., Line one <br> line two <br> line three <br> line four

Question sources

Computer Science Exit Exam practice questions · Question 29

Question 120 | Q01169

An ordered list is created with:

A.

B.

C.

D. None of the above

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer A:

Ordered lists are explicitly represented by ol on p.19. li marks items inside a list; ul is the unordered container. Evidence: PDF p. 19, PDF p. 19.

Why each choice is right or wrong

A - Correct: Exact ordered-list container.

B - Incorrect: Unordered-list container.

C - Incorrect: An item, not the ordered-list container.

D - Incorrect: A is documented.

Primary note proof

Study-note section WEB08

> Unordered <ul> and ordered <ol> contain <li> items.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 19 | lines 14-22

> II. Ordered Lists (ol) - element represents an ordered list of items - typically
> rendered as a numbered list.
> Format:
>
> ...
> ......
>
> Attribute: type="number type" {1, i, I, a, A}
> E.g. <ol type="i"> book
marker
chalk

Web Programming module | PDF page 19 | lines 2-5

```
<ul>
<li>...</li>
<li>...</li>...
</ul>
```

Question sources

Computer Science Exit Exam practice questions · Question 30

Question 121 | Q01170

Which markup indicates the copyright symbol in HTML?

A. &times

B. &copy

C. &reg

D. &lt

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer B: ©

WEB06 explicitly includes © among HTML entities. Thus the choice stem &copy identifies the copyright reference; the complete unambiguous spelling includes its semicolon. The other option names refer to different character references rather than copyright. This answer identifies the entity name, not a permission or copyright-policy assertion.

Why each choice is right or wrong

A - Incorrect: The times entity name does not identify the copyright symbol requested in the stem.

B - Correct: The copy entity name identifies copyright; write the full reference as © including the semicolon.

C - Incorrect: The reg entity name is not the copyright reference explicitly identified in WEB06.

D - Incorrect: The lt entity is the less-than reference, not the copyright reference.

Primary note proof

Study-note section WEB06

> Entities include ©, <, > and &.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 15 | lines 2-4

> 2.2.5 HTML Text Formatting tags
> In this section we will see the way HTML can be used to structure a page of text
> by adding headings, paragraphs, emphasizing words, creating lists and more

Question sources

Computer Science Exit Exam practice questions · Question 31

Question 122 | Q01171

In which HTML tag allows you to supply all kinds of information about your web page.

A. a | <head> x

B. i? <meta> v

C. /@| <title> x

D. | <body> x

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer B: i? <meta> v

The meta section defines meta elements as structured metadata about a web page. head contains metadata elements but is the broader container; title supplies only a title, body the visible content. Evidence: PDF p. 13, PDF p. 12.

Why each choice is right or wrong

A - Incorrect: Broader container, not the metadata element requested.

B - Correct: Exact metadata element.

C - Incorrect: Only supplies page title.

D - Incorrect: Contains visible page content.

Primary note proof

Study-note section WEB06

> Meta charset declares encoding; description summarizes the page; viewport controls mobile viewport behavior.

Study-note section WEB06

> <!DOCTYPE html> selects modern standards mode; <html> is the document root, <head> holds metadata, and <body> contains document content.

Study-note section WEB06

> <title> supplies the page/tab title.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 13 | lines 4-10

> 2.2.2 Meta tag
> Meta elements are tags used in HTML document to provide structured metadata about a web page. This element represents metadata that cannot be represented by

other HTML meta-related elements, like <base>, <link>, <script>, <style> or <title>. Meta elements are part of a web pages head section. There can be more than one meta element with different attributes on the same web page. In general, meta elements convey hidden information about the document.

Web Programming module | PDF page 12 | lines 6-20

<head>...</head>
- Contains information which is not displayed in the browser display area and may contain other tags in it
- Both the start and end tags may be omitted and inferred from child elements (HTML5)
- The head element may contain other elements such as base, link, meta, style, script, title
<title></title>
- Web browsers usually display it in a window's title bar when the window is open, and (where applicable) in the task bar when the window is minimized.
- Sets the title of the web page to be displayed in the browser's title bar and is found within the <head> tag.
<body></body>
-The body tag contains the visible part of the web page and is displayed in the

Question sources

Computer Science Exit Exam practice questions · Question 32

Question 123 | Q01180

Which of the following is used to add styles and formatting to a webpage in CSS?

A. Functions

B. Variables

C. Selectors

D. Arrays

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer C: Selectors

A selector identifies the elements to be styled and its declaration block supplies the styles (p.29). C is the listed mechanism for attaching styling rules to webpage elements. Evidence: PDF p. 29.

Why each choice is right or wrong

A - Incorrect: Functions are not the ruleset component selecting elements.

B - Incorrect: Variables are not the selecting component described.

C - Correct: Selectors identify the targets of CSS rules.

D - Incorrect: Arrays are not a CSS rule component here.

Primary note proof

Study-note section WEB11

> A rule is selector { property: value; }.

Study-note section WEB11

> A selector chooses where declarations apply; it is not itself the property that sets a color.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 29 | lines 8-15

> A selector group and an associated declarations block, together, are called a ruleset, or often a rule. A Cascading Style Sheet (CSS) rule is a statement that defines the style of one or more elements in your web page. These rules follow a specific structure.
> The whole structure is called a ruleset. Note the names of the individual parts:
> Selector -
> This is the HTML element name at the start of the ruleset. It defines the element(s) to be styled.

Question sources

Computer Science model exam 2 · Question 1

Question 124 | Q01200

Using PHP8 standard database extensions, which snippet can issue the shown SELECT against MySQL? Assume $pdo is an initialized PDO connection using the MySQL driver, $conn is an initialized MySQLi connection, and the users table exists.

A. $result = mysql_query("SELECT * FROM users WHERE id = 1");

B. $result=mysqliquery(conn, "SELECT * FROM users WHERE id = 1");

C. $result = $pdo->query("SELECT * FROM users WHERE id = 1");

D. $result = pg_query("SELECT * FROM users WHERE id = 1");

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer C: $result = $pdo->query("SELECT * FROM users WHERE id = 1");

With the stated MySQL PDO connection, $pdo->query executes the SELECT and returns its statement object, so C is valid. The old mysql_* extension is removed in PHP8. B misspells mysqli_query and omits the dollar prefix on its connection variable. pg_query belongs to the PostgreSQL extension and does not execute through either stated MySQL connection.

Why each choice is right or wrong

A - Incorrect: mysql_query belongs to the removed mysql_* extension and is not a PHP8 standard MySQL call.

B - Incorrect: The procedural spelling is mysqli_query($conn, ...); the offered name and bare conn do not form that call.

C - Correct: The initialized MySQL-driver PDO connection can execute this SELECT through its query method.

D - Incorrect: pg_query belongs to the PostgreSQL extension rather than the stated MySQL connections.

Primary note proof

Study-note section WEB31

> PHP8 uses MySQLi or PDO; the old mysql_* extension was removed, so migrate older snippets to a supported API.

Study-note section WEB-S04

> With an initialized PDO connection, $pdo->query($sql) executes an SQL statement without placeholders and returns a PDOStatement for processing its results.

Study-note section WEB-S04

> The procedural MySQLi query form is mysqli_query($conn, $sql), using a MySQLi connection and the correctly spelled function name.

Study-note section WEB-S04

> The pg_query function belongs to PHP's PostgreSQL extension; it is not a MySQL connection API.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 80 | lines 2-15

> 5.13.3 Sending Query to MySQL Database using PHP
> A database query from PHP is basically a MySQL command wrapped up in a tiny
> PHP function called mysql_query(). This is where you use the basic SQL workhorses
> of SELECT,
> INSERT, UPDATE, and DELETE, CREATE or DROP a table (but not those to create
> or drop an entire database) can be used with this PHP function.
> Syntax:
> [$resultvar=] mysql_query("query",[link identifier]);

$resultvar - is an optional result variable. This is used when the query is a select statement.
Query - is any valid insert, delete, update, select, create table or drop table statement enclosed by quotes. It is a good habit to hold the query in a separate variable to add flexibility to your code.
Link identifier-is an optional link identifier

Web Programming module | PDF page 80 | lines 16-23

5.13.4 Processing Query Result.
4. Fetching data sets
The result of a query returned by the mysql_query() function is an interger showing the success or failure of the function not the desired data. So you have to use one of the mysql_fetch_..() functions to get the actual result. There are many functions to do so, but the most commonly used ones are- Mysql_fetch_row($result) - this function takes the result of the mysql_query() function as an argument and returns the data as an enumerated array.

Question sources

Computer Science model exam 2 · Question 21

Question 125 | Q01201

Assume document.body exists and element is a newly created, detached div. Which snippet inserts that element into the document body?

A. element.appendChild(document.createTextNode("Hello, world!"));

B. var element = document.createElement("div");

C. document.body.appendChild(element);

D. document.getElementByld("container").appendChild(element);

Hint

Track which node is selected or created, and distinguish changing content from attaching a node.

Answer C: document.body.appendChild(element);

Creating a DOM element and inserting it are separate steps. The question supplies the already-created detached div, so appendChild on document.body performs the required insertion. Creating another div would leave it detached, and appending a text node changes the contents of an existing element rather than inserting that element into the body. The final choice retains the incorrect lookup spelling, which fails instead of providing the required insertion.

Why each choice is right or wrong

A - Incorrect: This adds a text node inside element, leaving element detached.

B - Incorrect: This creates another detached div rather than inserting it.

C - Correct: This attaches the supplied div to the existing document body.

D - Incorrect: getElementByld is misspelled and is not the documented lookup method.

Primary note proof

Study-note section WEB20

> Creation alone does not insert it.

Study-note section WEB20

> createTextNode makes text; appendChild attaches an existing node.

Study-note section WEB20

> getElementByld with a lowercase l replacing I is a different, nonexistent method.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 52 | lines 3-20

> 4.10 JavaScript DOM (Document object Model)
> The Document Object Model (DOM) is the data representation of the objects that comprise the structure and content of a document on the web. The Document Object Model (DOM) is a programming interface for web documents. It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects; that way, programming languages can interact with the page.
> 4.10.1 Accessing HTML elements in JavaScript
> In the DOM Api, there are different methods to get access to HTML elements. We will list the available methods here, but only the most recent methods are covered here.
> - getElementById(id) returns an element with that specific ID
> - getElmentsByClassName(class) returns elements with that specific class
> getElementsByTagName(tag) returns elements with the given tag name
> - querySelector(selector) → The Document method querySelector() returns the first Element within the document that matches the specified selector, or group of selectors. If no matches are found, null is returned.
> element = document.querySelector(selectors);

Question sources

Computer Science model exam 2 · Question 22

Question 126 | Q01221

Which of the following is used to include an external JavaScript file in an HTML document?

A. <script href="file.js"></script>

B. <link rel="stylesheet" href="file.js">

C. <script type="text/javascript" src="file.js"></script>

D. <link rel="stylesheet" type="text/javascript" href="file.js">

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer C: <script type="text/javascript" src="file.js"></script>

The module external-file syntax uses a script element with src pointing to script.js. C uses this structure; stylesheet link elements do not load executable script in that example. Evidence: PDF p. 45, PDF p. 29.

Why each choice is right or wrong

A - Incorrect: href is not the script source attribute shown.

B - Incorrect: Links as stylesheet instead of script.

C - Correct: Uses script and src as required.

D - Incorrect: Still links as a stylesheet rather than script.

Primary note proof

Study-note section WEB15

> Use <script src="app.js"></script> for an external .js file; script content has no HTML closing-tag syntax inside its JavaScript source.

Study-note section WEB11

> An external sheet uses <link rel="stylesheet" href="styles.css"> in head; an internal sheet uses style; an inline declaration uses an element’s style attribute.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 45 | lines 12-14

> - We can also include JavaScript as an external file. We can use a similar syntax
> like the internal and add the src attribute. Here is how
> <script src="script.js" defer></script>

Web Programming module | PDF page 29 | lines 2-6

> E.g.
> <link rel="stylesheet" href="style.css">

The rel attributes names the relationship of the linked document to the current document. The href attribute specifies the URL of the linked document. The URL can be relative or absolute

Question sources

Computer Science model exam 2 · Question 42

Question 127 | Q01224

Which of the following is used to define the structure of a webpage in HTML?

A. Tags

B. Classes

C. IDs

D. Styles

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer A: Tags

HTML tags delimit elements that establish content structure (pp.10-11). Classes and IDs identify elements, while styles concern presentation. Evidence: PDF p. 10, PDF p. 11.

Why each choice is right or wrong

A - Correct: Tags form structural elements.

B - Incorrect: Identifiers do not replace structural tags.

C - Incorrect: Unique identifiers do not replace structural tags.

D - Incorrect: Presentation rather than the HTML structural mechanism.

Primary note proof

Study-note section WEB06

Tags mark boundaries; an element includes content and attributes, or is a void element such as img.

Study-note section WEB11

Type p selects paragraph elements, .notice selects a class, #logo an ID, h1,h2 groups selectors, nav ul selects descendants and nav>ul direct children.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 10 | lines 9-16

HTML (HyperText Markup Language) is the most basic building block of the Web. HTML is not a programming language rather it is a markup language that tells web browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Web Programming module | PDF page 11 | lines 11-17

2.2 HTML Tags
In HTML, a tag is used for creating an element. Tags are used to delimit the start and end of elements in the markup. Normal html elements have a start tag to indicate where they begin, and an end tag to indicate where they end. The start and end tags of certain normal elements can be omitted. An HTML tag is composed of the name of the element, surrounded by angle brackets (<>). An end tag also has a slash after the opening angle bracket, to distinguish it from the start tag.

Question sources

Computer Science model exam 2 · Question 45

Question 128 | Q01235

Which PHP mechanisms can participate in generating dynamic content and server-side processing? Select all that apply.

A. Functions

B. Variables

C. Classes

D. Objects

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A, B, C, D: Functions; Variables; Classes; Objects

Dynamic PHP responses can combine values, reusable behavior and object-based structures. Variables hold incoming or computed data; functions carry out processing; classes define application state and behavior; objects are instances used by that logic. None of these mechanisms alone defines dynamic PHP, but each can participate in it. The revised wording therefore asks for all contributors rather than forcing one exclusive mechanism from four compatible choices.

Why each choice is right or wrong

A - Correct: Functions can process inputs and return values or produce output.

B - Correct: Variables hold request data and computed values used in a response.

C - Correct: Classes define state and behavior for server application logic.

D - Correct: Objects instantiate classes and can participate in generating responses.

Primary note proof

Study-note section WEB26

A function can receive arguments, return a value, or produce output; classes/objects/variables also support server processing, so no single one of these mechanisms alone defines “dynamic PHP.” Use braces to make nesting explicit.

Study-note section WEB33

A class defines properties/methods; new creates an object, $obj->field accesses a property, and $obj->method() calls a method.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 4-16

PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.
Client-side scripting is the glamorous, eye-catching part of Web development. In contrast, server-side scripting is invisible to the user. Server-side Web scripting is mostly about connecting Web sites to back-end servers, such as databases. This enables the following types of two-way communication:
➢ Server to client: Web pages can be assembled from back end-server output.
➢ Client to server: Customer-entered information can be acted upon.

Web Programming module | PDF page 71 | lines 23-30

5.9 PHP Functions
A function is a set of statements that performs a particular function and-optionally-returns a value. Functions have many advantages
- Involve less typing
- Reduce syntax and other programming errors
- Decrease the loading time of program files
PHP comes with hundreds of ready-made functions. You have seen some of them like print, echo, count, sizeof

Web Programming module | PDF page 85 | lines 13-18

Object-oriented programming (OOP) opens the door to cleaner designs, easier maintenance, and greater code reuse. In php OOP we use the term class, which refers

to a template for building objects. And an object is an instance of a class. The data associated with an object are called its properties. The functions associated with an object are called its methods. When you define a class, you define the names of its properties and give the code for its methods.

Web Programming module | PDF page 62 | lines 3-10

Variables are used to store data, like string of text, numbers, etc. Variable values can change over the course of a script. Here're some important things to know about variables:
- In PHP, a variable does not need to be declared before adding a value to it. PHP automatically converts the variable to the correct data type, depending on its value.
- After declaring a variable, it can be reused throughout the code.
- The assignment operator (=) used to assign value to a variable.

Question sources

Computer Science model exam 2 · Question 56

Question 129 | Q01236

Which of these JavaScript libraries, frameworks or platforms can contribute to interactive web interfaces when combined with suitable responsive layout and styling? Select all that apply.

A. jQuery

B. React

C. Angular

D. Vue.js

Hint

Match the stated application requirement to the capabilities and category of each technology.

Answer A, B, C, D: jQuery; React; Angular; Vue.js

jQuery supports DOM operations, React builds component interfaces, and Angular and Vue also support interactive interfaces. Their categories differ, but the revised question includes libraries, frameworks and platforms rather than treating those terms as exclusive. Responsive behavior additionally requires appropriate layout and styling; it is not guaranteed by choosing one name. Each listed technology can contribute under that condition, so all four choices are valid.

Why each choice is right or wrong

A - Correct: jQuery can provide DOM interaction while layout rules handle responsive presentation.

B - Correct: React components can implement interactive interfaces with responsive styles.

C - Correct: Angular supplies component-based application capabilities.

D - Correct: Vue can build interactive interfaces and use responsive layout styles.

Primary note proof

Study-note section WEB38

> jQuery simplifies DOM operations; React builds component interfaces; Angular is a component platform, distinct from older AngularJS; Vue also builds interactive interfaces.

Study-note section WEB38

> Each can contribute to a responsive interface, so those generic descriptions do not identify one unique framework.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 94 | lines 14-17

> d. React.js
> React.js is an open-source front-end JavaScript library for building user interfaces based on UI components. React builds encapsulated component that manage their own state, then compose them to complex UIs. To build for the web, developers use

Web Programming module | PDF page 93 | lines 12-19

> c. Angular.js
> Angular is a development platform, built on TypeScript. As a platform, Angular includes:
> 1. A component-based framework for building scalable web applications
> 2. A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more
> 3. A suite of developer tools to help you develop, build, test, and update your code

Web Programming module | PDF page 91 | lines 28-31

> 2. Many of Bootstrap components require the use of JavaScript to function.
> Specifically, they require jQuery, Popper.js, and boostrap JavaScript plugins.
> Follow this order when including the JS files
> - JQuery

Question sources

Computer Science model exam 2 · Question 57

Question 130 | Q01239

Which snippet applies a one-column card layout only when the viewport is at most 600 CSS pixels wide?

A. @media (max-width: 600px) { .cards { grid-template-columns: 1fr; } }

B. .container { display: flex; }

C. box { transform: rotate(45deg); }

D. .menu { background-color: #f00; }

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer A: @media (max-width: 600px) { .cards { grid-template-columns: 1fr; } }

The required behavior depends on viewport width, so the rule must contain the appropriate media condition and a concrete layout declaration. max-width:600px applies at or below that width, and the single 1fr track makes the card grid one column. A plain flex declaration does not express this breakpoint or column requirement. Rotation and background color affect appearance without implementing the requested conditional layout.

Why each choice is right or wrong

A - Correct: The media condition and one-track declaration implement the stated breakpoint behavior.

B - Incorrect: This does not specify the requested viewport condition or card-grid track count.

C - Incorrect: Rotation does not create the conditional one-column layout.

D - Incorrect: A background color does not alter the grid at the required breakpoint.

Primary note proof

Study-note section WEB14

> Responsive design combines flexible layout, images, viewport metadata and media queries, for example @media(max-width:600px){.cards{grid-template-columns:1fr}}.

Study-note section WEB14

> Grid organizes rows and columns with explicit tracks, such as grid-template-columns:1fr 2fr.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 37 | lines 15-18

> Display
> The display CSS property sets whether an element is treated as a block or inline element and the layout used for its children, such as flow layout, grid or flex.
> Formally, the display property sets an element's inner and outer display types. The

Web Programming module | PDF page 91 | lines 6-13

> Now a days the website you build has to support all browser types and all sizes of screen (Desktop, Tablets, Phones etc..). To make that possible we use different frameworks. This part of the chapter will introduce you to the Bootstrap framework.
> Bootstrap is a free front end framework for faster and easier web development.
> It is s an open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

Web Programming module | PDF page 34 | lines 4-10

> The background shorthand CSS property sets all background style properties at once, such as color, image, origin and size, or repeat method. You have also specific background properties that you can apply like background-color, background-image etc.
> To set the background-color of an element and provide the value of the color in different form.
> 1. Keyword values -> red, blue

Question sources

Computer Science model exam 2 · Question 60

Question 131 | Q01244

Which of the following is used to store user data on the server-side in web applications?

A. Cookies

B. Sessions

C. Local storage

D. IndexedDB

Hint

Follow the identifier and server-side state through the stated lifecycle or access check; distinguish client-controlled claims from server decisions.

Answer B: Sessions

In conventional server-backed sessions, the browser normally keeps an opaque identifier and the user’s session data resides in server-controlled storage. Cookies, localStorage and IndexedDB instead retain data on the client side. Client storage may participate in an application design, but it is not the server-side session-data store requested. Sessions therefore match the stated location.

Why each choice is right or wrong

A - Incorrect: Cookies hold browser-side values; a session cookie normally carries an identifier rather than the server’s session-data store.

B - Correct: A server-backed session keeps the user's session data in server-controlled storage.

C - Incorrect: localStorage retains origin-specific strings in browser storage rather than storing the server's session data.

D - Incorrect: IndexedDB is structured local browser storage rather than the server's session-data store.

Primary note proof

Study-note section WEB29

> Normally the browser holds an opaque session identifier while data resides in server-controlled storage.

Study-note section WEB28

> Cookies are browser-held name/value data automatically attached to matching requests subject to domain, path, expiry, Secure and SameSite rules.

Study-note section WEB28

> localStorage persists origin-specific string values across sessions until removed/cleared; sessionStorage is scoped to a page session/tab.

Study-note section WEB28

> IndexedDB supplies structured local storage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 76 | lines 2-5

> in one application. A PHP session allows you to store user information on the server for later use (i.e., username, shopping items, etc.). However, session information is temporary and will be deleted after the user has left the website. If you need a permanent storage you may want to store the data in a database.

Web Programming module | PDF page 58 | lines 12-19

> Cookies let you store user information in web pages.
> What are Cookies?
> Cookies are data, stored in small text files, on your computer.
> When a web server has sent a web page to a browser, the connection is shut down, and the server forgets everything about the user. Cookies were invented to solve the problem "how to remember information about the user":
> When a user visits a web page, his/her name can be stored in a cookie.
> Next time the user visits the page, the cookie "remembers" his/her name.

Question sources

Computer Science model exam 2 · Question 65

Question 132 | Q01515

Identify the function in JavaScript which is used to send messages to users requesting for text input?

A. getInput()

B. alert()

C. prompt ()

D. display()

Hint

Distinguish the browser object or dialog role from the kind of response it collects.

Answer C: prompt ()

Prompt displays a modal asking for text input; alert is an informational popup. C names the documented text-input operation. Evidence: PDF p. 46, PDF p. 47.

Why each choice is right or wrong

A - Incorrect: Not the documented input function.

B - Incorrect: Informational popup, not text entry.

C - Correct: Exact text-input function.

D - Incorrect: Not the documented input function.

Primary note proof

Study-note section WEB22

> alert displays information, confirm returns a Boolean, and prompt requests text or returns null on cancellation.

Study-note section WEB-S03

> Alert is a notification; confirm additionally asks for an OK/Cancel decision, while prompt additionally offers a text field.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 46 | lines 16-20

> Accepting user input in JavaScript actually is not a difficult task. JavaScript has a function called prompt, which is used to accept input from the user. But for the result to be useful for us, it has to be stored in a variable. Here is a simple example
> var name = prompt("Enter your name: ");
> alert("Hello " + name);

Web Programming module | PDF page 47 | lines 2-2

> 2. Alert: shows a browser pop up with the information inside the parenthesis

Question sources

Mock exam 1 · Question 31

Question 133 | Q01516

In HTML, which attribute is used to create a link that opens in a new window tab?

A. src="_blank"

B. target="_self"

C. target="_blank"

D. src="_self"

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C: target="_blank"

WEB07 says _blank targets a new browsing context, while _self uses the current one. Therefore target="_blank" is the offered attribute-value pair for a new tab or window. Browser preferences determine whether the new context appears as a tab or window, so the markup does not force that UI distinction. src does not select a link target context.

Why each choice is right or wrong

A - Incorrect: src does not select the browsing context where an anchor link is opened.

B - Incorrect: The _self target uses the current context rather than requesting a new one.

C - Correct: The _blank target requests a new context; browser preferences decide whether that is a tab or window.

D - Incorrect: src is not the target attribute, and _self would indicate the current context anyway.

Primary note proof

Study-note section WEB07

> target="_self" uses the current context, _blank a new context, _parent its parent and _top the top-level context.

Study-note section WEB07

> Browser preferences determine tab versus window.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 14 | lines 5-9

> In HTML, a link is created by wrapping the text or other content inside an <a> tag (anchor tag) and using the href attribute, also known as Hypertext reference, or target

that contains the web address.
Here is an example of a link to the google website
Google.

Web Programming module | PDF page 21 | lines 29-31

1. application/x-www-form-urlencoded → most common and used from simple scenarios
2. mutlipart/form-data → used to upload binary data

Question sources

Mock exam 1 · Question 32

Question 134 | Q01517

In which HTML element, we put the JavaScript code using?

A. <javascript>...</javascript>

B. <script>...</script>

C. <js>...</js>

D. <css>...</css>

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B: <script>...</script>

Page 45 encloses JavaScript between script and /script. None of the invented javascript, js or css element names matches that example. Evidence: PDF p. 45.

Why each choice is right or wrong

A - Incorrect: Not the shown element.

B - Correct: Exact enclosing element.

C - Incorrect: Not the shown element.

D - Incorrect: CSS is presentation and this is not the script element.

Primary note proof

Study-note section WEB15

> Use <script src="app.js"></script> for an external .js file; script content has no HTML closing-tag syntax inside its JavaScript source.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 45 | lines 2-8

> <script>
> // JavaScript code goes here
> </script>
> - Internal JavaScript is one way of including JavaScript using the script tag inside our HTML document. We can use the above syntax to include it anywhere in our document. The best place to include a JavaScript file is before the end of body tag </body>.

Question sources

Mock exam 1 · Question 33

Question 135 | Q01609

Which of the following CSS style property is used to specify an italic text?

A. font-weight

B. font-family

C. font-size

D. font-style

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer D: font-style

The font property list distinguishes font-style, font-weight, font-family and font-size; its example uses italic (pp.34-35). The style component corresponds to italic, rather than weight, family or size. Evidence: PDF p. 34, PDF p. 35.

Why each choice is right or wrong

A - Incorrect: Weight concerns heaviness, not the style component named in the list.

B - Incorrect: Family names the typeface, not italic style.

C - Incorrect: Size sets dimensions rather than italic style.

D - Correct: Style is the named italic-related font component.

Primary note proof

Study-note section WEB13

> color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 34 | lines 21-31

> 3.3.2 Font and Text Properties
> The font CSS shorthand property sets all the different properties of an element's font. Alternatively, it sets an element's font to a system font. The property is a shorthand for the following properties
> - font-family
> - font-size
> - font-stretch
> - font-style
> - font-variant

- font-weight
- line-height

Web Programming module | PDF page 35 | lines 2-2

font: italic 1.2em "Fira Sans", serif;

Question sources

Mock exam 3 · Question 41

Question 136 | Q01610

Which of the following function is used to create a session?

A. isset()

B. $_SESSION()

C. session_destroy()

D. session_start()

Hint

Follow the identifier and server-side state through the stated lifecycle or access check; distinguish client-controlled claims from server decisions.

Answer D: session_start()

The Starting a PHP Session example calls session_start() before storing data. $_SESSION stores data; session_destroy removes it; isset tests presence. Evidence: PDF p. 76, PDF p. 77, PDF p. 73.

Why each choice is right or wrong

A - Incorrect: Presence test, not startup.

B - Incorrect: SESSION is the storage array, not initializer function.

C - Incorrect: Destroys instead of starting.

D - Correct: Exact session startup function.

Primary note proof

Study-note section WEB29

session_start() starts/resumes PHP session handling and makes $_SESSION available.

Study-note section WEB-S09

session_destroy() removes the data associated with the current PHP session.

Study-note section WEB27

isset tests existence and non-null, not truth or valid syntax.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 76 | lines 6-23

Starting a PHP Session
Before you can store user information in your PHP session, you must first start up the session.
<?php session_start(); ?>
<html>
<body>.....
</body>
</html>
The code above will register the user's session with the server, allow you to start saving user information, and assign a UID for that user's session.
Storing a Session Variable
The correct way to store and retrieve session variables is to use the PHP $_SESSION variable:
<?php
session_start();
// store session data
$_SESSION['views']=1;

Web Programming module | PDF page 77 | lines 5-16

Destroying a Session
If you wish to delete some session data, you can use the unset() or the session_destroy() function. The unset() function is used to free the specified session variable:
<?php
unset($_SESSION['views']);
?>
You can also completely destroy the session by calling the session_destroy() function:
<?php
session_destroy();
?>
session_destroy() will reset your session and you will lose all your stored session data.

Web Programming module | PDF page 73 | lines 9-10

• isset(): This function is used to determine whether the variable or a form control is having a value or not.

Question sources

Mock exam 3 · Question 42

Question 137 | Q01611

Which of the following is the correct output for the following JavaScript code:
function check(option) {
return(option ? "true" : "false");
}
bool ans=true;
console.log(check(ans));

A. false

B. Runtime Error

C. true

D. Compilation Error

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer D: Parse-time syntax error, described by the option as Compilation Error.

WEB19 explicitly states that bool ans=true is not a JavaScript declaration. Parsing fails before check(ans) is called, so neither string result is logged. Replacing bool with let would make the conditional return "true", but that is a different program. D is the closest offered category for the source's parse-time failure rather than a runtime exception.

Why each choice is right or wrong

A - Incorrect: The conditional function is never executed because the declaration prevents successful parsing.

B - Incorrect: The failure occurs during parsing, before a running program could raise a runtime error.

C - Incorrect: true would follow only after repairing bool to a valid JavaScript declaration keyword.

D - Correct: Compilation Error is the offered label closest to the actual parse-time SyntaxError.

Primary note proof

Study-note section WEB19

> bool ans=true is not JavaScript declaration syntax; use let ans=true.

Study-note section WEB-S05

> A syntax error prevents source from being parsed successfully.

Study-note section WEB17

> A condition ? x : y chooses one branch.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 48 | lines 6-22

4.5.2 Variable declaration in JavaScript
Variables are used to store this information. A variable is a “named storage” for data. We can use variables to store goodies, visitors, and other data. We have three ways of creating variables in JavaScript
- Using the let keyword → the following statement declares a variable
let message;
now we can put values using the assignment operator (=)
- Using var keyword → we can also declare variable using the var keyword in a similar fashion as let. There is a subtle difference between declaring a variable using var and let. Variables declared with var are hoisted where as those declared with let are not. The other difference between var and let is variables declared with let has a block level scope unlike that of var. Variables declared with var have function level scope
- Using const keyword → to declare a constant variable we can use const instead of var and let. Variables declared with const can not be changed once they are declared.
const PI = 3.14;

Web Programming module | PDF page 50 | lines 15-18

Ternary Operators:
Ternary operator is represented by a question mark “?”. We can use ternary operator to perform conditional operations like if-else statement
let result = condition ? value1 : value2;

Question sources

Mock exam 3 · Question 43

Question 138 | Q01651

What makes AJAX unique?

A. It works the same with all web browsers

B. It works as a stand-alone web-development tool

C. It uses C++ as its programming language

D. It makes data requests asynchronously

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer D: It makes data requests asynchronously

AJAX is expressly asynchronous and exchanges data with the server without refreshing the page. It combines XML, HTML, CSS and JavaScript rather than being a standalone language/tool. Evidence: PDF p. 89.

Why each choice is right or wrong

A - Incorrect: All-browser identity is not the stated distinguishing feature.

B - Incorrect: Contradicts use of several cooperating technologies.

C - Incorrect: The documented scripting language is JavaScript, not C++.

D - Correct: Exact asynchronous-request distinction.

Primary note proof

Study-note section WEB34

> AJAX means Asynchronous JavaScript and XML, but responses can be JSON, text, HTML or XML.

Study-note section WEB34

> It is a technique, not a language, standalone framework or mandatory XML protocol.

Study-note section WEB02

> Open standards do not guarantee identical support for every feature in every browser.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 4-13

> AJAX is a web development technique for creating interactive web applications.
> AJAX stands for Asynchronous JavaScript and XML. It can send and receive information in various formats, including JSON, XML, HTML, and text files. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script. AJAX's most appealing characteristic is its "asynchronous" nature, which means it can communicate with the server, exchange data, and update the page without having to refresh the page.
> The two major features of AJAX allow you to do the following:
> - Make requests to the server without reloading the page
> - Receive and work with data from the server

Question sources

Mock exam 2 · Question 40

Question 139 | Q01652

Which of the following attribute is used for merging two or more adjacent columns?

A. COLSPAN

B. CELLPADDING

C. CELLSPACING

D. ROWSPAN

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer A: COLSPAN

Page 18 says colspan and rowspan span columns and rows respectively. Merging adjacent columns requires colspan. Evidence: PDF p. 18.

Why each choice is right or wrong

A - Correct: Columns are explicitly the colspan role.

B - Incorrect: Not the merging attribute identified.

C - Incorrect: Not the merging attribute identified.

D - Incorrect: Spans rows instead of columns.

Primary note proof

Study-note section WEB08

> colspan spans columns; rowspan spans rows.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 18 | lines 15-19

> If you have a reason to merge columns or rows or the website you are building needs this functionality you can use the colspan and rowspan attribute to do so. These attributes are used to span two or more columns or rows respectively. Both colspan and rowspan takes unitless numbers that specifies the number of columns and rows you want to span

Question sources

Mock exam 2 · Question 41

Question 140 | Q01653

Which of the following CSS selector is used to specify a rule to bind a particular unique element?

A. Class

B. Tag

C. Id

D. Name

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer C: Id

The module ID selector example targets a specific element, whereas type and class selectors can target groups. C matches the unique identifier requirement. Evidence: PDF p. 31.

Why each choice is right or wrong

A - Incorrect: Class selection can group elements.

B - Incorrect: Type selection applies to all elements of that type.

C - Correct: ID selects the particular identified element.

D - Incorrect: Name is not the unique ID selector described.

Primary note proof

Study-note section WEB11

> Type p selects paragraph elements, .notice selects a class, #logo an ID, h1,h2 groups selectors, nav ul selects descendants and nav>ul direct children.

Study-note section WEB07

> Attributes belong in opening tags, with unique IDs when used for document identification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 31 | lines 5-24

> 2. Class selector -> The element(s) on the page with the specified class. Multiple
> instances of the same class can appear on a page. We use
> the . (dot) symbol and the name of the class to use it as a
> selector in css
> 3. Id selector -> The element on the page with the specified ID. On a given
> HTML page, each id value should be unique. We use # (hashtag
>) symbol and the name of the id as a selector
> 4. Attribute Selector -> The element(s) on the page with the specified
> attribute.
> 5. Pseudo-class selector -> The specified element(s), but only when in the

specified state. (For example, when a cursor hovers over
a link.)
Q. Read in detail the above ways of selectors it's definition, advantage, syntax and examples for each?
3.2.1 Style Inheritance
In CSS, inheritance controls what happens when no value is specified for a property on an element. CSS uses the document tree for inheritance, in which an element applied to an element is inherited by its descendants.
CSS properties can be categorized in two types:
- Inherited properties - which by default are set to the computed value of

Question sources

Mock exam 2 · Question 42

Question 141 | SM0005

Which statement about ordinary JavaScript functions and identifiers is true?

A. Function names can contain spaces

B. Function names are not case-sensitive

C. Functions can be passed to and stored by other code like other values

D. Function names can begin with digits

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer C: Functions can be passed to and stored by other code like other values

JavaScript functions are values that can be passed as arguments or stored for later use. Identifier rules still apply to ordinary function names: names are case-sensitive, cannot contain spaces and cannot begin with a digit. The true choice tests a supported function property instead of making an unrestricted claim about the number of parameters. Runtime argument limits therefore do not affect the correctness of this version.

Why each choice is right or wrong

A - Incorrect: Spaces are not permitted inside an ordinary function identifier.

B - Incorrect: Identifier names are case-sensitive.

C - Correct: Functions can be stored and passed as values.

D - Incorrect: An ordinary function identifier cannot start with a digit.

Primary note proof

Study-note section WEB15

Identifiers are case-sensitive and may start with letters, underscore or $, not digits or spaces.

Study-note section WEB19

function f(x){return x*2;} has a parameter and returns a value; functions can be passed/stored like other values.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 51 | lines 18-35

4.9 JavaScript Functions
Functions are the main “building blocks” of the program. They allow the code to be called many times without repetition. To create a function, we can use a function declaration.
function showCSMessage() {
alert('Hello CS Students!');
}
To create a function, first we use the function keyword then the function name.
There are times when we remove the function name. This type Functions without a function name are called anonyms functions.
After the function name we optionally put parameters in between the parenthesis, if we don’t have parameters, we always have to put the empty parenthesis and then finally the block of code to be executed.
An arrow function expression is a compact alternative to a traditional function expression, but is limited and can't be used in all situations.
Syntax:
param => expression
Multiple params require parentheses.

Question sources

MOE Computer Science Exit Exam · Question 2-1

National Computer Science Exit Exam · Question 4

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 20

Question 142 | SM0014

Look at the following PHP line: $list = array(20, 12, 23, 19, 18); What is the index of data value 19?

A. 2

B. 0

C. 3

D. 1

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: 3

PHP numeric array examples start at index zero. In the ordered list, 20 is index 0, 12 is 1, 23 is 2, 19 is 3 and 18 is 4. Evidence: PDF p. 69, PDF p. 70.

Why each choice is right or wrong

A - Incorrect: Index 2 holds 23.

B - Incorrect: Index 0 holds 20.

C - Correct: Index 3 holds 19.

D - Incorrect: Index 1 holds 12.

Primary note proof

Study-note section WEB26

> Numeric arrays start at index0 unless keys are supplied: [20,12,23,19,18] has19 at index3.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 69 | lines 14-15

```
<?php
$array1 = array("Item 1", "Item 2", "Item 3", "Item 4");
```

Web Programming module | PDF page 70 | lines 2-8

```
We can also create arrays by adding the index of the item in between the square
brackets. This will enable us to add items in unorderly fashion
<?php
$sample[0] = "sample 1";
$sample[1] = "sample 2";
$sample[2] = "sample 3";
$sample[3] = "sample 4";
```

Question sources

MOE Computer Science Exit Exam · Question 4-2

National Computer Science Exit Exam · Question 18

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 21

Question 143 | SM0039

Which option is a complete, valid PHP variable-assignment statement inside an open PHP block? No closing PHP tag follows the option to supply an implicit terminator.

A. $test = 9.5

B. $test = "This is 'Computer science'";

C. $test = "Computer;

D. test = 5;

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B: $test = "This is 'Computer science'";

The balanced double-quoted assignment is complete: it has a dollar-prefixed variable, a matching closing quote and a semicolon. Single quotes inside double quotes are ordinary string characters. The numeric assignment lacks its required semicolon in the stated context, another option has an unterminated string, and the bare test name is not a PHP variable.

Why each choice is right or wrong

A - Incorrect: This is an assignment expression but lacks the semicolon required for a complete statement in the stated context.

B - Correct: The variable name, balanced double quotes and terminating semicolon make a complete assignment.

C - Incorrect: The opening double quote has no matching closing delimiter.

D - Incorrect: A PHP variable must have its dollar prefix; bare test does not name that variable.

Primary note proof

Study-note section WEB23

> Variables begin with $, then a letter/underscore and subsequent letters/digits/underscores; $AGE18 works but $18AGE does not.

Study-note section WEB-S06

> A complete ordinary PHP assignment statement ends with a semicolon.

Study-note section WEB-S06

A string’s closing delimiter must match its opening delimiter; a single quote can appear inside a double-quoted string without ending it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 62 | lines 27-33

In PHP user defined variable can be declared as:
$var_name = value;
Example
<?php
// Declaring variables
$txt = "Hello World!";
$number = 10;

Web Programming module | PDF page 64 | lines 6-12

All variables in PHP are denoted with a leading dollar sign ($).
All variables in PHP start with a leading $ sign. After the initial $, variable names must be composed of letters (uppercase or lowercase), digits (0-9), and underscore characters (_). Furthermore, the first character after the $ may not be a number.
➢ The value of a variable is the value of its most recent assignment.
➢ Variables are assigned with the = operator, with the variable on the left-hand side and the expression to be evaluated on the right.

Web Programming module | PDF page 61 | lines 28-29

echo "Hello World!";
?>

Question sources

MOE Computer Science Exit Exam · Question 12-1

National Computer Science Exit Exam · Question 53

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 22

Question 144 | SM0068

Which one of the following is the smallest heading tag?

A. <H6>

B. <H3>

C. <H4>

D. <H1>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer A: <H6>

The module explicitly calls h1 largest and h6 smallest among the default heading formats. A therefore identifies the smallest default heading. Evidence: PDF p. 15.

Why each choice is right or wrong

A - Correct: h6 is explicitly smallest.

B - Incorrect: h3 is not the smallest.

C - Incorrect: h4 is not the smallest.

D - Incorrect: h1 is explicitly largest.

Primary note proof

Study-note section WEB06

> Heading levels h1-h6 express hierarchy, with h6 conventionally smallest in default styling, not a mandatory CSS size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 15 | lines 5-10

> Headings -
> Headings are predefined formats for text presentation and there are six heading formats defined in HTML:
> <h1> up to <h6>
> <h1> the largest font size
> <h6> the smallest font size

Question sources

MOE Computer Science Exit Exam · Question 19-4

National Computer Science Exit Exam · Question 44

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 23

Question 145 | SM0070

In PHP 8 with short_open_tag enabled, which shown delimiters do NOT open and close PHP code? Select all that apply.

A. <? ..... ?>

B. <caption> ..... </caption>

C. <script language="PHP"> ..... </script>

D. <?php ..... ?>

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B, C: <caption> ..... </caption>; <script language="PHP"> ..... </script>

The full <?php form is valid, and the stated configuration also permits the short <? form. A caption element is HTML markup rather than a PHP delimiter. The historical script-language form is no longer accepted in PHP 8, so it is another non-delimiter under this version. Specifying both version and short-tag configuration makes B and C definite selections rather than relying on an unstated historical environment.

Why each choice is right or wrong

A - Incorrect: The explicitly enabled short-tag configuration makes this PHP delimiter valid.

B - Correct: caption is HTML markup, not a PHP code delimiter.

C - Correct: PHP 8 does not accept this historical script-language delimiter.

D - Incorrect: The full PHP opening and closing form is valid.

Primary note proof

Study-note section WEB23

> Use <?php ...

Study-note section WEB23

> PHP8 no longer accepts historical <script language="PHP"> delimiters; short <? depends on configuration, so use the full form.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 61 | lines 5-14

> Short-open (SGML-style) tags
> Short or short-open tags look like this:
> <?
> //php script
> ?>

Those who escape into and out of HTML frequently in each script will be attracted by the prospect of fewer keystrokes; however, the price of shorter tags is pretty high. Additional adjustment in the php.ini file is required to enable PHP to recognize the tags. PHP code written with short-open tags is less portable because you can‘t be sure another machine will have enabled them in the php.ini file.

Web Programming module | PDF page 60 | lines 23-27

5.2 Basic PHP Syntax
The most universally effective PHP syntax is:
<?php
//php script
?>

Web Programming module | PDF page 18 | lines 21-22

<table>
<caption align=“center” valign=“bottom”>table 1.0</caption>

Question sources

MOE Computer Science Exit Exam · Question 20-2

National Computer Science Exit Exam · Question 73

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 24

Question 146 | SM0077

Which statements about Java and JavaScript are true? Select all that apply.

A. Both are object oriented languages

B. JavaScript and Java are the same programming language

C. Javascript can be used in extremely sophisticated applications

D. JavaScript function identifiers are not case-sensitive

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer A, C: Both are object oriented languages; Javascript can be used in extremely sophisticated applications

Java supports class-based object-oriented programming, while JavaScript supports prototype-based objects; both therefore support object-oriented programming. The Java classification is supplied by the supplementary Oracle tutorial, while the note supplies JavaScript's object model, its distinction from Java and its ability to support sophisticated applications. The languages are not identical, and JavaScript identifiers remain case-sensitive. Thus both the object-oriented statement and the sophisticated-application capability are true.

Why each choice is right or wrong

A - Correct: Both support object-oriented programming, with different object models.

B - Incorrect: The note explicitly distinguishes JavaScript from Java.

C - Correct: The note explicitly supports sophisticated JavaScript applications.

D - Incorrect: JavaScript identifiers are case-sensitive.

Primary note proof

Study-note section WEB-S01

> Java organizes object-oriented programs using classes, objects, inheritance and interfaces.

Study-note section WEB15

> JavaScript originated at Netscape; it is distinct from Java and supports sophisticated applications through prototype-based objects and first-class functions.

Study-note section WEB15

> Identifiers are case-sensitive and may start with letters, underscore or $, not digits or spaces.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 4-19

> JavaScript (JS) is a lightweight, interpreted or compiled programming language. It is a single threaded, dynamic language, supporting object oriented, functional programming styles. Do not confuse JavaScript with the Java Programming language. The two-programming language have very different syntax, semantics, and use.
> 4.2 JavaScript Basic
> JavaScript is a scripting or programming language that allows us to implement complex features on web pages. JavaScript will enable us to display timely content updates, interactive maps, 2D/3D animated graphics etc. JavaScript is one of the three layers of web technologies which lets to add functionality to a web page. The three layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'
> What can JavaScript really do? It can
> - Stores values inside variables

- Perform operations on pieces of texts
- Run a code in response to events

Question sources

MOE Computer Science Exit Exam · Question 22-2

National Computer Science Exit Exam · Question 29

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 25

Question 147 | SM0080

Which one of the following is true about frames in HTML?

A. Every frame must display the same HTML document

B. A legacy frameset document uses body in place of frameset

C. All browsers support frames

D. Frames allow parts of the page to remain stationary while other parts scroll

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer D: Frames allow parts of the page to remain stationary while other parts scroll

A legacy frameset divides a window into separate documents, and individual frame views can scroll independently. One frame can therefore remain stationary while another scrolls. The documents need not be identical, and the old frameset document used frameset in place of body. The noframes fallback addresses browsers without frame support, so support cannot be asserted for all browsers.

Why each choice is right or wrong

A - Incorrect: Separate frames can display different documents; identical content is not required.

B - Incorrect: The old frameset document replaced body with frameset, the reverse of this claim.

C - Incorrect: The noframes fallback exists for browsers without frame support, so universal support is false.

D - Correct: Independent scrolling lets one frame remain visible while another view scrolls.

Primary note proof

Study-note section WEB10

Legacy frameset/frame divides a window into separate documents; frameset replaced body in old frameset documents.

Study-note section WEB10

Named targets controlled which frame changed; independent scrolling could leave other frames stationary.

Study-note section WEB-S07

A legacy frameset document could include noframes content for user agents that did not support frames or were configured not to display them.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 20 | lines 4-15

2.2.9 HTML Frames
HTML frames allows us to present documents in multiple views, which may be independent windows or sub-windows. Multiple views offer designers a way to keep certain information visible, while other views are scrolled or replaced.
An HTML document that describes frame layout has a different markup than HTML document without frames. A standard document has one HEAD section and one BODY. A frameset document has a HEAD and a FRAMESET in place of BODY.
2.2.9.1 Frame Set
The FRAMESET section of a document specifies the layout of views in the main user agent window. In addition, the FRAMESET section can contain a NOFRAMES element to provide alternate content for user agents that do not support frames or are configured not to display frames.

Question sources

MOE Computer Science Exit Exam · Question 23-1

National Computer Science Exit Exam · Question 76

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 26

Question 148 | SM0085

What will be the output of the following JavaScript statement? Math.floor(5.9)

A. 9

B. 11.8

C. 6

D. 5

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer D: 5

WEB18 directly evaluates Math.floor(5.9)=5. The floor operation gives the greatest integer not exceeding the input, rather than rounding to the nearest integer or multiplying the input. Thus the fractional part does not push the result up to six; the expression value is five. No string or DOM behavior changes this math result.

Why each choice is right or wrong

A - Incorrect: Nine is not the integer floor of 5.9; it is merely a digit appearing in the input.

B - Incorrect: 11.8 doubles the input and is unrelated to the floor operation.

C - Incorrect: Six would be upward or nearest-integer rounding here, not floor.

D - Correct: Five is the greatest integer no larger than 5.9, matching the explicit note example.

Primary note proof

Study-note section WEB-S08

> For finite numeric arguments, Math.abs(x) gives the nonnegative magnitude, Math.floor(x) gives the greatest integer no larger than x, and Math.sqrt(x) gives the nonnegative square root for nonnegative x.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 84 | lines 17-21

> 5.16 PHP Mathematical Functions
> The math functions can handle values within the range of integer and float types.The PHP math functions are part of the PHP core. No installation is required to use these functions.
> Here are some PHP math functions, you can get the whole list from the internet
> abs - Absolute value

Web Programming module | PDF page 85 | lines 2-5

> base_convert - Convert a number between arbitrary bases
> bindec - Binary to decimal
> ceil - Round fractions up
> cos - Cosine

Question sources

MOE Computer Science Exit Exam · Question 24-2

National Computer Science Exit Exam · Question 81

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 27

Question 149 | SM0099

What will be the output of the following JavaScript statement? Math.sqrt(36)

A. 6

B. 30

C. 1

D. 36

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer A: 6

WEB18 explicitly states Math.sqrt(36)=6. Six squared gives thirty-six, and this function returns the nonnegative square root rather than the original operand or an arithmetic difference. The expression is already a valid numeric function call and needs no parsing repair or context-dependent type coercion to obtain the stated result.

Why each choice is right or wrong

A - Correct: Six is the nonnegative square root of thirty-six, exactly as WEB18 evaluates it.

B - Incorrect: Thirty is not a square root of thirty-six and does not match the defined function.

C - Incorrect: One squared is one, not thirty-six, so it cannot be the result.

D - Incorrect: Thirty-six is the input; sqrt does not simply return it unchanged.

Primary note proof

Study-note section WEB-S08

> For finite numeric arguments, Math.abs(x) gives the nonnegative magnitude, Math.floor(x) gives the greatest integer no larger than x, and Math.sqrt(x) gives the nonnegative square root for nonnegative x.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 84 | lines 17-21

> 5.16 PHP Mathematical Functions
> The math functions can handle values within the range of integer and float types.The PHP math functions are part of the PHP core. No installation is required to use these functions.
> Here are some PHP math functions, you can get the whole list from the internet
> abs - Absolute value

Question sources

MOE Computer Science Exit Exam · Question 28-2

National Computer Science Exit Exam · Question 13

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 29

Question 150 | SM0215

How are the objects organized in the HTML DOM?

A. Hierarchy

B. Queue

C. Stack

D. Class-wise

Hint

Track which node is selected or created, and distinguish changing content from attaching a node.

Answer A: Hierarchy

The module describes a document tree with parent elements and descendants (p.31), and DOM nodes/objects representing document structure (p.52). These relations form a hierarchy rather than a queue, stack or class grouping. Evidence: PDF p. 31, PDF p. 52.

Why each choice is right or wrong

A - Correct: Tree parent/descendant structure is hierarchical.

B - Incorrect: No FIFO sequence is stated as the document organization.

C - Incorrect: No LIFO sequence is stated as the document organization.

D - Incorrect: Class selectors group matches, not the document parent/child structure.

Primary note proof

Study-note section WEB20

> The DOM is a hierarchy of nodes representing the document.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 31 | lines 19-22

> 3.2.1 Style Inheritance
> In CSS, inheritance controls what happens when no value is specified for a

property on an element. CSS uses the document tree for inheritance, in which an element applied to an element is inherited by its descendants.

Web Programming module | PDF page 52 | lines 3-9

4.10 JavaScript DOM (Document object Model)
The Document Object Model (DOM) is the data representation of the objects that comprise the structure and content of a document on the web. The Document Object Model (DOM) is a programming interface for web documents. It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects; that way, programming languages can interact with the page.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 5-2

Computer Science Exit Exam practice collection · Question 40

Question 151 | SM0232

Which of the following methods can be used to refresh a webpage?

A. location.relocate

B. window.relocate

C. window.refresh

D. location.reload

Hint

Distinguish the browser object or dialog role from the kind of response it collects.

Answer D: location.reload()

WEB22 explicitly says location.reload() reloads the current page. D names that method; it must be called with parentheses to perform the reload. The other alternatives invent relocate or refresh names not identified as the standard method in the note. This operation acts on the current browsing location rather than creating a new page context.

Why each choice is right or wrong

A - Incorrect: location.relocate is not the reload method identified in the browser API rule.

B - Incorrect: window.relocate is not the reload method given in WEB22.

C - Incorrect: window.refresh is not the standard reload method described by the supplied note.

D - Correct: Calling location.reload() reloads the current page; the option names that method.

Primary note proof

Study-note section WEB22

location.reload() reloads the current page.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 58 | lines 2-10

The window.location object can be written without the window prefix.
Some examples:
window.location.href returns the href (URL) of the current page
window.location.hostname returns the domain name of the web host
window.location.pathname returns the path and filename of the current page
window.location.protocol returns the web protocol used (http: or https:)
window.location.assign() loads a new document
Window Location Href
The window.location.href property returns the URL of the current page.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 10-3

Computer Science Exit Exam practice collection · Question 41

Question 152 | SM0255

Which one of the following is a false statement?

A. Aweb browser is software that runs on a server

B. Awebsite is a collection of resources in various forms

C. Aweb page is the basic unit of information storage on the www

D. Auser agent renders resources for a user to view

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Aweb browser is software that runs on a server

The module places page storage on servers and page viewing with browsers on clients (p.9). A reverses that teaching role. B is consistent with hosted website files; C is explicit web-information storage; D matches the viewing role. Evidence: PDF p. 9, PDF p. 9, PDF p. 8.

Why each choice is right or wrong

A - Correct: False in the module client/server model.

B - Incorrect: Consistent with the collection of hosted website files.

C - Incorrect: Explicitly web information resides in pages.

D - Incorrect: Consistent with browser viewing/rendering resources.

Primary note proof

Study-note section WEB01

A browser/user agent requests resources and renders HTML/CSS, then runs permitted JavaScript.

Study-note section WEB01

A web/HTTP server receives requests and returns representations, sometimes invoking application code.

Study-note section WEB02

A website groups resources; a page is one document/resource.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 9 | lines 2-4

• Web pages are files stored on computers called Web servers.
• Computers reading the Web pages are called Web clients.
• Web clients view the pages with a program called a Web browser.

Web Programming module | PDF page 9 | lines 13-21

Web hosting is the place where all the files of your website live. Web hosting is a service that allows organizations and individuals to post a website or a web page onto the Internet. A web host, or web hosting service provider, is a business that provides the technologies and services needed for the website or webpage to be viewed in the Internet. Websites are hosted, or stored, on special computers called servers. When Internet users want to view your website, all they need to do is type your website address or domain (the domain name for the site has to be registered) into their browser. Their computer will then connect to your server and your webpages will be delivered to them through the browser.

Web Programming module | PDF page 8 | lines 34-34

• Web information is stored in documents called Web pages.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 18-1

Question 153 | SM0267

Which one of the following terms refers to the machine on which the requested resource is stored?

A. Port

B. Query

C. Host

D. Protocol

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: Host

The web hosting section says website files reside on a host/server; the location examples identify hostname as the web-host domain. Host therefore names the resource-storing machine role. Evidence: PDF p. 9, PDF p. 58, PDF p. 8.

Why each choice is right or wrong

A - Incorrect: Not the hosting computer role defined.

B - Incorrect: A database/request operation does not name the hosting machine.

C - Correct: Host matches the storage/serving machine role.

D - Incorrect: HTTP is a communication standard rather than a storage machine.

Primary note proof

Study-note section WEB02

> A URL identifies scheme, host, optional port, path, query and fragment; the host identifies the machine/service name, while a fragment selects a local document location.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 9 | lines 13-21

> Web hosting is the place where all the files of your website live. Web hosting
> is a service that allows organizations and individuals to post a website or a web page
> onto the Internet. A web host, or web hosting service provider, is a business that
> provides the technologies and services needed for the website or webpage to be
> viewed in the Internet. Websites are hosted, or stored, on special computers called
> servers. When Internet users want to view your website, all they need to do is type
> your website address or domain (the domain name for the site has to be registered)
> into their browser. Their computer will then connect to your server and your webpages
> will be delivered to them through the browser.

Web Programming module | PDF page 58 | lines 4-7

> window.location.href returns the href (URL) of the current page
> window.location.hostname returns the domain name of the web host
> window.location.pathname returns the path and filename of the current page
> window.location.protocol returns the web protocol used (http: or https:)

Web Programming module | PDF page 8 | lines 32-32

> • All the computers use a communication standard called HTTP.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 22-1

Computer Science Exit Exam practice collection · Question 191

Question 154 | SM0281

Which attribute of the <img> tag is used to display text in place of an image when the image is not displayed?

A. rep

B. pla

C. alt

D. im

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C: alt

The img alt value is explicitly a textual description shown when the image cannot be seen or takes too long to render. Evidence: PDF p. 16.

Why each choice is right or wrong

A - Incorrect: Not the named fallback attribute.

B - Incorrect: Not the named fallback attribute.

C - Correct: Exact documented fallback attribute.

D - Incorrect: Not the named fallback attribute.

Primary note proof

Study-note section WEB07

> <img src="photo.jpg" alt="Students in a lab"> supplies source and alternative text.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 16 | lines 27-29

> Another attribute we frequently see in an img tag is alt. Its value is textual description of the image, which is displayed in a situation where the image cannot be seen or takes a long time to render.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 26-3

Computer Science Exit Exam practice collection · Question 36

Question 155 | SM0284

Which of the following is the top-level object in web browsers?

A. Root object

B. Document object

C. JavaScript object

D. Window object

Hint

Distinguish the browser object or dialog role from the kind of response it collects.

Answer D: Window object

The BOM discussion states that global JavaScript objects/functions/variables belong to window and even document is its property. Thus window is the top-level browser object in this teaching model. Evidence: PDF p. 56.

Why each choice is right or wrong

A - Incorrect: Generic root label is not the documented browser object name.

B - Incorrect: Document is a property within window.

C - Incorrect: Generic JavaScript object is not the named top-level object.

D - Correct: Window contains the global members and document.

Primary note proof

Study-note section WEB22

> In a normal browser page, window exposes the browsing context and document is its document property.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 56 | lines 15-25

> The window object is supported by all browsers. It represents the browser's window.
> All global JavaScript objects, functions, and variables automatically become members of the window object.
> Global variables are properties of the window object
> Global functions are methods of the window object.
> Even the document object (of the HTML DOM) is a property of the window object:
> window.document.getElementById("header");
> is the same as:
> document.getElementById("header");

Question sources

Computer Science Exit Exam · Sene 2016 · Question 27-3

Computer Science Exit Exam practice collection · Question 37

Question 156 | SS095

Which CSS selector selects elements whose class list contains notice?

A. p

B. #notice

C. [type="email"]

D. .notice

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer D: .notice

A class selector uses a dot followed by the class name, so .notice matches elements that have notice among their classes. An ID selector uses a hash and tests a different attribute. The remaining choices are a type selector and an attribute selector, each meaningful but aimed at another condition. The damaged options are replaced with explicit selector forms rather than claiming to recover an uncertain source-letter mapping.

Why each choice is right or wrong

A - Incorrect: This selects paragraph elements by tag name.

B - Incorrect: This selects the notice ID, not the class token.

C - Incorrect: This tests the type attribute rather than the class list.

D - Correct: This is the class selector for the notice token.

Primary note proof

Study-note section WEB11

> Type p selects paragraph elements, .notice selects a class, #logo an ID, h1,h2 groups selectors, nav ul selects descendants and nav>ul direct children.

Study-note section WEB11

> Attribute selectors match properties such as [type="email"], pseudo-classes states such as :hover/:focus, and pseudo-elements parts such as ::before.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 31 | lines 5-11

2. Class selector -> The element(s) on the page with the specified class. Multiple instances of the same class can appear on a page. We use the . (dot) symbol and the name of the class to use it as a selector in css
3. Id selector -> The element on the page with the specified ID. On a given HTML page, each id value should be unique. We use # (hashtag) symbol and the name of the id as a selector

Question sources

Computer Science model exam collection · Question 2

Question 157 | SS101

Which option identifies components present in an HTTP/1.1 request even when it has no message body?

A. Header fields only

B. Header fields and a mandatory body

C. A response status line, header fields and a mandatory body

D. A request line and header fields

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer D: A request line and header fields

HTTP/1.1 requests start with a request line, then contain header fields and a terminating blank line. A body is optional, so neither option that requires one describes every request. A status line begins a response, not a request. The version qualifier matters because HTTP/2 and HTTP/3 encode messages differently; the selected components refer specifically to the textual HTTP/1.1 structure, not every HTTP wire format.

Why each choice is right or wrong

A - Incorrect: This omits the request line.

B - Incorrect: The body is optional, so making it mandatory is wrong.

C - Incorrect: A status line belongs to a response, and a request body is optional.

D - Correct: These components are present before any optional body.

Primary note proof

Study-note section WEB03

An HTTP/1.1 request begins with method, request-target and version, followed by header fields, a blank line, and an optional body.

Study-note section WEB03

A response begins with version, status code and optional reason phrase, then headers, blank line and permitted content.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 20-31

6.3 Sending Request to PHP server
XMLHttpRequest (XHR) is an API that can be used by JavaScript and other web browser scripting languages to transfer and manipulate XML data to and from a webserver using HTTP, establishing an independent connection channel between a webpage's Client-Side and Server-Side.
In order to make an HTTP request to the server with JavaScript, you need an instance of an XMLHttpRequest Object. We make a request to the sever, by calling the open() and send() methods of the HTTP request object, like this:
httpRequest.open('GET', 'http://www.example.org/some.file', true);
httpRequest.send();
open() takes three parameters, the first is the HTTP request method and the second is the URL. The optional third parameter sets whether the request is asynchronous.

Question sources

Computer Science model exam collection · Question 8

Question 158 | SS131

What will be the output of the following code snippet?
<script type="text/javascript" language= 'javascript' >
var a = "exitexam";
var result = a.substring(2, 6);
document.write(result);
</script>

A. xitex

B. texam

C. itex

D. xite

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer C: itex

substring(2,6) uses a zero-based start and excludes the end index. In exitexam, indices 2, 3, 4 and 5 contain i, t, e and x. document.write therefore emits itex. Starting at index 1 or continuing beyond index 5 produces a different interval, which rules out the other strings.

Why each choice is right or wrong

A - Incorrect: xitex starts at index one and contains five characters, rather than the requested four-character interval.

B - Incorrect: texam starts too late and extends beyond the exclusive end index six.

C - Correct: Indices two through five contain i, t, e, x, producing itex.

D - Incorrect: xite begins at index one rather than the specified zero-based start index two.

Primary note proof

Study-note section WEB18

> A JS string’s substring(start,end) includes start and excludes end: "exitexam".substring(2,6) is "itex".

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 16-19

> What can JavaScript really do? It can
> - Stores values inside variables
> - Perform operations on pieces of texts
> - Run a code in response to events

Web Programming module | PDF page 47 | lines 24-31

> - String → a string in JavaScript must be surrounded by quotes. We have three different quotes
> 1. Single quotes --- ‘single’
> 2. Double quotes --- ‘double’
> 3. Backticks -- `hello`. In a standard US keyboard, you can find the backticks above the tab key before the number 1 key.
> Backticks allows to embed variables and expressions into a string by wrapping them with ${}

Question sources

Computer Science model exam collection · Question 38

Question 159 | SS181

Default method while submitting a form is

A. Get method

B. Set method

C. Post method

D. Put method

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer A: GET

WEB09 explicitly says a form's default method is GET. Therefore omitting the method attribute does not select POST or PUT. WEB04 further distinguishes GET query parameters from POST form bodies, but that difference does not change the default. Set is not the relevant form submission method named in the note.

Why each choice is right or wrong

A - Correct: GET is the default method for an HTML form when its method attribute is omitted.

B - Incorrect: Set is not the default HTML form submission method defined here.

C - Incorrect: POST must be requested explicitly; it is not the form's omitted-method default.

D - Incorrect: PUT is not the default method selected by an ordinary HTML form.

Primary note proof

Study-note section WEB09

> The default method is GET.

Study-note section WEB04

> GET forms usually put name/value pairs in the URL query; POST forms put them in the body.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 21 | lines 23-31

> Form attributes:
> - action → Backend script ready to process the data collected
> - method → method used to upload Data. The most frequently used
> - target → Specify the target window or frame where the result of the script
> will be displayed. It takes values like _blank, _self, _parent etc.
> - enctype → tells how the browser encodes data. Possible values
> 1. application/x-www-form-urlencoded → most common and used from
> simple scenarios
> 2. mutlipart/form-data → used to upload binary data

Web Programming module | PDF page 62 | lines 23-26

> - $_GET → An associative array of variables passed to the current script via the
> HTTP GET method.
> - $_POST → An associative array of variables passed to the current script via
> the HTTP POST method.

Question sources

Computer Science model exam collection · Question 88

Question 160 | Q01135_LOCAL69

A variable declared _____ has a LOCAL SCOPE?

A. outside program

B. inside function

C. outside function

D. None of the Above

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: Inside a function

WEB25 states that a function's locals and by-value parameters are separate bindings from variables outside it. Thus a variable declared inside a function has ordinary local scope, unless a special explicit global binding is introduced. Outside function is the global-scope location, while outside program is not a meaningful declaration location in the script.

Why each choice is right or wrong

A - Incorrect: Outside program does not describe a declaration location within the executing PHP script.

B - Correct: An ordinary declaration inside a function creates a local binding for that function.

C - Incorrect: Variables declared outside functions have global scope under WEB25.

D - Incorrect: Inside function is offered and matches the ordinary local-binding rule.

Primary note proof

Study-note section WEB25

> A function's locals and by-value parameters are separate bindings; use global $x or $GLOBALS["x"] to refer to a global explicitly.

Study-note section WEB25

> A variable outside a function has global scope.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 62 | lines 15-18

> All the following variables are automatically available in every scope.
> - $GLOBALS → Contains a reference to every variable which is currently available within the global scope of the script. The keys of this array are the names of the global variables.

Web Programming module | PDF page 72 | lines 7-17

Function definitions have the following form:
function function_name([parameter [, ...]])
{
// Statements
}
That is, function definitions have four parts:
- A definition starts with the word function.
- A name follows, which must start with a letter or underscore, followed by any number of letters, numbers, or underscores.
- The parentheses are required.
- One or more parameters, separated by commas, are optional

Question sources

Web Programming practice questions · Question 69

Question 161 | AM0051

What is the primary markup language used for structuring content on the web?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: HTML

HTML is explicitly a markup language telling browsers how to structure pages, while CSS handles appearance and JavaScript behavior. Evidence: PDF p. 10, PDF p. 60.

Why each choice is right or wrong

A - Correct: Exact markup/structure role.

B - Incorrect: Stylesheet/presentation role.

C - Incorrect: Programming/behavior role.

D - Incorrect: Server-side scripting role.

Primary note proof

Study-note section WEB01

HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 10 | lines 9-16

HTML (HyperText Markup Language) is the most basic building block of the Web.
HTML is not a programming language rather it is a markup language that tells web browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Web Programming module | PDF page 60 | lines 4-10

PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science MCQ practice collection · Question 51

Question 162 | AM0052

What is the purpose of CSS in web development?

A. To define the semantic document structure using HTML elements

B. To add interactivity and dynamic behavior to web pages

C. To control the visual appearance of web pages

D. To handle server-side processing and database interactions

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: To control the visual appearance of web pages

CSS controls visual presentation, including colors, typography and layout. HTML supplies document structure and meaning; JavaScript supplies programmable browser interaction. Server technologies such as PHP generate responses and interact with databases. With the structural distractor confined to semantic elements, the presentation alternative is the unique description of CSS's role.

Why each choice is right or wrong

A - Incorrect: Semantic document elements and structure are HTML's role rather than CSS's presentation role.

B - Incorrect: Programmable interaction is the JavaScript role, distinct from CSS styling.

C - Correct: Visual presentation is the central CSS role.

D - Incorrect: Server processing and database access belong to server application code.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 26 | lines 4-16

> In the previous chapter we have seen how to structure a web page using HTML.
> Now it is time to add presentation to our sites, to do that we will be using a technology called CSS. CSS (Cascading Style Sheets) is the code that styles web content. Using CSS you can apply styles to make web pages look exactly how you want. This works because CSS is connected to the DOM(Document Object Model). CSS is not a programming language. It's not a markup language either. CSS is a style sheet language. CSS is what you use to selectively style HTML elements.
> p {
> color: red;
> }
> 3.1.1 Introduction to CSS
> CSS stands for Cascading Style Sheet. It is a stylesheet language used to describe the presentation of a document written in HTML. Styles define how to display HTML

Web Programming module | PDF page 10 | lines 14-16

> HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Web Programming module | PDF page 60 | lines 11-16

> Client-side scripting is the glamorous, eye-catching part of Web development. In contrast, server-side scripting is invisible to the user. Server-side Web scripting is mostly about connecting Web sites to back-end servers, such as databases. This enables the following types of two-way communication:
> ➢ Server to client: Web pages can be assembled from back end-server output.
> ➢ Client to server: Customer-entered information can be acted upon.

Question sources

Computer Science MCQ practice collection · Question 52

Question 163 | AM0053

Which programming language is primarily used for client-side scripting in web development?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: JavaScript

The client-side scripting chapter identifies JavaScript as the language adding browser behavior; HTML structures, CSS presents, PHP executes server-side. Evidence: PDF p. 44, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: Markup structure, not the scripting language.

B - Incorrect: Stylesheet presentation, not scripting.

C - Correct: Exact browser scripting language.

D - Incorrect: Server-side language in p.60.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 2-15

Chapter 4: Client-Side Scripting (JavaScript)
4.1 Introduction to JavaScript
JavaScript (JS) is a lightweight, interpreted or compiled programming language. It is a single threaded, dynamic language, supporting object oriented, functional programming styles. Do not confuse JavaScript with the Java Programming language. The two-programming language have very different syntax, semantics, and use.
4.2 JavaScript Basic
JavaScript is a scripting or programming language that allows us to implement complex features on web pages. JavaScript will enable us to display timely content updates, interactive maps, 2D/3D animated graphics etc. JavaScript is one of the three layers of web technologies which lets to add functionality to a web page. The three layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'

Web Programming module | PDF page 60 | lines 4-10

PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science MCQ practice collection · Question 53

Question 164 | AM0054

Which application-layer protocol defines the requests and responses used to retrieve web resources?

A. HTTP

B. FTP

C. TCP/IP

D. SMTP

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: HTTP

HTTP defines the application-level exchange for web resource requests and responses. TCP/IP supplies underlying communication facilities, so naming the application layer distinguishes the required role. FTP and SMTP concern other application services rather than the web resource exchange described. HTTPS adds TLS protection to HTTP; the question asks for the underlying web application protocol.

Why each choice is right or wrong

A - Correct: HTTP defines the web resource request/response exchange.

B - Incorrect: FTP is a file-transfer protocol, not the application protocol defining the stated web resource exchange.

C - Incorrect: TCP/IP names underlying network/transport facilities, not this application-layer request/response protocol.

D - Incorrect: SMTP concerns mail transfer rather than the web resource request/response exchange.

Primary note proof

Study-note section WEB01

> The Internet connects networks; the Web is a resource system commonly using HTTP.

Study-note section WEB-S14

> HTTP defines web application requests, response status codes and representations.

Study-note section WEB-S14

> An HTTP exchange can use underlying TCP connections and IP addresses; naming that network stack does not identify the application-level method or status semantics.

Study-note section WEB-S14

> FTP is a file-transfer application protocol.

Study-note section WEB-S14

> SMTP transports electronic mail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 8 | lines 29-34

> • The World Wide Web (WWW) is most often called the Web.
> • The Web is a network of computers all over the world.
> • All the computers in the Web can communicate with each other.

• All the computers use a communication standard called HTTP.
How does the WWW work?
• Web information is stored in documents called Web pages.

Question sources

Computer Science MCQ practice collection · Question 54

Question 165 | AM0055

What is the purpose of a web server in web development?

A. To store and manage website files

B. To handle client-side scripting

C. To control the visual appearance of web pages

D. To ensure secure data transmission over the web

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: To store and manage website files

Web servers store the files comprising websites and deliver pages to requesting browsers (p.9). Client interactivity and visual styling have separate JavaScript/CSS roles. Evidence: PDF p. 9, PDF p. 44.

Why each choice is right or wrong

A - Correct: Direct storage/hosting role.

B - Incorrect: Client-side behavior is assigned to JavaScript.

C - Incorrect: Presentation is assigned to CSS.

D - Incorrect: Secure transport is not the general server definition supplied.

Primary note proof

Study-note section WEB01

HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 9 | lines 13-21

> Web hosting is the place where all the files of your website live. Web hosting is a service that allows organizations and individuals to post a website or a web page onto the Internet. A web host, or web hosting service provider, is a business that provides the technologies and services needed for the website or webpage to be viewed in the Internet. Websites are hosted, or stored, on special computers called servers. When Internet users want to view your website, all they need to do is type your website address or domain (the domain name for the site has to be registered) into their browser. Their computer will then connect to your server and your webpages will be delivered to them through the browser.

Web Programming module | PDF page 44 | lines 13-15

> layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'

Question sources

Computer Science MCQ practice collection · Question 55

Question 166 | AM0056

Which listed DBMS is the open-source relational SQL server used by the course's PHP database examples, including its MySQLi connection and query operations?

A. MySQL

B. Oracle

C. SQL Server

D. MongoDB

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer A: MySQL

The note identifies MySQL as an open-source SQL DBMS and uses it as the database server for the PHP MySQLi examples. The question asks for that specific course example, so MySQL is the matching system. Oracle, SQL Server and MongoDB are different named systems; this selection does not depend on an unsupported popularity ranking or a claim that no other database has open-source editions.

Why each choice is right or wrong

A - Correct: MySQL is the open-source SQL DBMS used as the server in the course's MySQLi examples.

B - Incorrect: Oracle is a different named database system, not the MySQL server used by these examples.

C - Incorrect: SQL Server is a different named database system from the server used by the course's MySQLi examples.

D - Incorrect: MongoDB is not the named SQL server used by the course's MySQLi examples.

Primary note proof

Study-note section WEB-S04

> MySQL is an open-source SQL database management system.

Study-note section WEB31

> MySQL is a relational DBMS; a database persists structured application data and supports concurrent clients.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 78 | lines 22-31

> 5.13 Database Programming using PHP
> 5.13.1 PHP File Overview on MySQL database
> MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. It is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.
> - MySQL is a database management system.
> A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network.

Question sources

Computer Science MCQ practice collection · Question 56

Question 167 | AM0057

what is the purpose of a cookie in web development?

A. To store and retrieve user preferences and session information

B. To control the visual appearance of web pages

C. To handle client-side scripting

D. To ensure secure data transmission over the web

Hint

Follow the identifier and server-side state through the stated lifecycle or access check; distinguish client-controlled claims from server decisions.

Answer A: To store and retrieve user preferences and session information

Cookies remember user information across requests and are stored as name/value pairs (p.58). A matches retaining preferences/session information rather than styling, executing scripts or guaranteeing transport security. Evidence: PDF p. 58, PDF p. 59.

Why each choice is right or wrong

A - Correct: Matches remembering user information.

B - Incorrect: Presentation is CSS, not cookie storage.

C - Incorrect: Cookies store data, not the client scripting language itself.

D - Incorrect: The cookie definition supplies no secure-transport guarantee.

Primary note proof

Study-note section WEB28

> Cookies are browser-held name/value data automatically attached to matching requests subject to domain, path, expiry, Secure and SameSite rules.

Study-note section WEB29

> Normally the browser holds an opaque session identifier while data resides in server-controlled storage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 58 | lines 12-23

> Cookies let you store user information in web pages.
> What are Cookies?
> Cookies are data, stored in small text files, on your computer.
> When a web server has sent a web page to a browser, the connection is shut down, and the server forgets everything about the user. Cookies were invented to solve the problem "how to remember information about the user":
> When a user visits a web page, his/her name can be stored in a cookie.
> Next time the user visits the page, the cookie "remembers" his/her name.
> Cookies are saved in name-value pairs like:
> When a browser requests a web page from a server, cookies belonging to the page are added to the request. This way the server gets the necessary data to "remember" information about users.

Web Programming module | PDF page 59 | lines 4-7

> Create a Cookie with JavaScript
> JavaScript can create, read, and delete cookies with the document.cookie property.
> With JavaScript, a cookie can be created like this:
> document.cookie = "username=John Doe";

Question sources

Computer Science MCQ practice collection · Question 57

Question 168 | AM0058

Which listed programming languages can execute server-side web application code? Select all that apply.

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C, D: JavaScript; PHP

PHP is a server-response language, and JavaScript can execute on the server through runtimes such as Node.js. JavaScript is therefore not excluded merely because it also runs in browsers. HTML supplies document structure and CSS supplies presentation rules rather than the requested server application scripting role. Both supported programming-language choices must be selected under the revised multiple-selection wording.

Why each choice is right or wrong

A - Incorrect: HTML supplies document structure and meaning rather than server application scripting.

B - Incorrect: CSS supplies presentation rules rather than server application scripting.

C - Correct: JavaScript can run server-side, including through Node.js.

D - Correct: PHP can generate server responses and access application data.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 4-10

> PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Web Programming module | PDF page 92 | lines 6-9

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.

Web Programming module | PDF page 10 | lines 9-16

> HTML (HyperText Markup Language) is the most basic building block of the Web. HTML is not a programming language rather it is a markup language that tells web browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
> HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Question sources

Computer Science MCQ practice collection · Question 58

Question 169 | AM0059

What is the purpose of a framework in web development?

A. To ensure secure data transmission over the web

B. To control the visual appearance of web pages

C. To provide a foundation and structure for developing web applications

D. To handle client-side scripting

Hint

Match the stated application requirement to the capabilities and category of each technology.

Answer C: To provide a foundation and structure for developing web applications

The framework chapter describes integrated components, libraries and developer tools for building applications. This provides the reusable foundation in C; particular frameworks may also style or script pages, but those are narrower uses. Evidence: PDF p. 93, PDF p. 91.

Why each choice is right or wrong

A - Incorrect: Secure transport is not the general framework purpose given.

B - Incorrect: Styling is a subset of Bootstrap functionality, not the general foundation role.

C - Correct: Matches integrated application-building structure/tools.

D - Incorrect: Client scripting is only one part of the framework examples.

Primary note proof

Study-note section WEB38

> A framework supplies conventions and reusable structure; a library supplies callable capabilities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 93 | lines 12-19

> c. Angular.js
> Angular is a development platform, built on TypeScript. As a platform, Angular includes:
> 1. A component-based framework for building scalable web applications
> 2. A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more
> 3. A suite of developer tools to help you develop, build, test, and update your code

Web Programming module | PDF page 91 | lines 10-15

> Bootstrap is a free front end framework for faster and easier web development.
> It is s an open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.
> Bootstrap solves many problems that we face when using HTML,CSS and vanilla JavaScript.

Question sources

Computer Science MCQ practice collection · Question 59

Question 170 | AM0060

Which programming language is often used for building scalable and high-performance web applications?

A. HTML

B. CSS

C. JavaScript

D. Node.js

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer C: JavaScript is the language; Node.js is a runtime.

WEB01 explicitly distinguishes Node.js from the language it runs. JavaScript is therefore the offered programming-language answer for Node-based server applications. HTML and CSS supply structure and presentation. Scalability and performance depend on workload and architecture, as WEB36 explains, so the answer identifies the language without promising that its use alone guarantees high performance.

Why each choice is right or wrong

A - Incorrect: HTML is structural markup rather than the server application language requested.

B - Incorrect: CSS supplies presentation rather than programmable server application logic.

C - Correct: JavaScript is the programming language used by Node applications.

D - Incorrect: Node.js is a JavaScript runtime, not a separate programming language.

Primary note proof

Study-note section WEB01

> Node.js is a JavaScript runtime, not a separate language.

Study-note section WEB36

> One event-loop thread can handle many waiting requests, but long CPU loops or synchronous I/O block its callbacks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 92 | lines 6-14

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.
> Benefits of using Node
> 1. Performance
> 2. Uses plain old JavaScript
> 3. Thousands of reusable packages
> 4. Portable

Web Programming module | PDF page 10 | lines 9-16

> HTML (HyperText Markup Language) is the most basic building block of the Web.
> HTML is not a programming language rather it is a markup language that tells web

browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Question sources

Computer Science MCQ practice collection · Question 60

Question 171 | AM0061

What is the purpose of AJAX in web development?

A. To control the visual appearance of web pages

B. To handle client-side scripting

C. To ensure secure data transmission over the web

D. To enable asynchronous communication between the client and server

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer D: To enable asynchronous communication between the client and server

AJAX asynchronously communicates with the server, exchanges data and updates pages without refresh (p.89). D is the precise purpose; client-side scripting is the broader environment, not the distinguishing function. Evidence: PDF p. 89.

Why each choice is right or wrong

A - Incorrect: Visual styling is a cooperating CSS role.

B - Incorrect: Broad surrounding activity, not the specific AJAX purpose.

C - Incorrect: No secure-transport guarantee is given.

D - Correct: Exact asynchronous client/server exchange purpose.

Primary note proof

Study-note section WEB34

AJAX means Asynchronous JavaScript and XML, but responses can be JSON, text, HTML or XML.

Study-note section WEB34

It is a technique, not a language, standalone framework or mandatory XML protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 4-13

> AJAX is a web development technique for creating interactive web applications.
> AJAX stands for Asynchronous JavaScript and XML. It can send and receive information in various formats, including JSON, XML, HTML, and text files. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script. AJAX's most appealing characteristic is its "asynchronous" nature, which means it can communicate with the server, exchange data, and update the page without having to refresh the page.
> The two major features of AJAX allow you to do the following:
> - Make requests to the server without reloading the page
> - Receive and work with data from the server

Question sources

Computer Science MCQ practice collection · Question 61

Question 172 | AM0062

Which listed formats can represent structured application data for server/browser interchange, rather than document presentation or styling? Select all that apply.

A. HTML

B. CSS

C. JSON

D. XML

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer C, D: JSON; XML

JSON represents structured values such as objects and arrays, while XML is another supported interchange format. A REST API may return either, so data interchange alone does not choose one uniquely. HTML can be delivered as a response document, but the revised question distinguishes that presentation role from the structured-data formats sought here. CSS expresses styling rules. JSON and XML therefore form the requested selection.

Why each choice is right or wrong

A - Incorrect: HTML's role here is the response document, not the requested structured-data interchange format.

B - Incorrect: CSS expresses presentation rules rather than application-data interchange.

C - Correct: JSON represents structured data values for interchange.

D - Correct: XML is another valid structured-data interchange format.

Primary note proof

Study-note section WEB35

> JSON represents objects, arrays, strings, numbers, Booleans and null; keys/strings use double quotes, and functions/undefined are not JSON values.

Study-note section WEB35

> XML is another valid interchange format.

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 4-10

> AJAX is a web development technique for creating interactive web applications.
> AJAX stands for Asynchronous JavaScript and XML. It can send and receive information in various formats, including JSON, XML, HTML, and text files. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script. AJAX's most appealing characteristic is its "asynchronous" nature, which means it can communicate with the server, exchange data, and update the page without having to refresh the page.

Question sources

Computer Science MCQ practice collection · Question 62

Question 173 | AM0063

What is the purpose of a content management system (CMS) in web development?

A. To store and manage website files

B. To control the visual appearance of web pages

C. To handle client-side scripting

D. To provide a user-friendly interface for managing website content

Hint

Distinguish the purpose of each development or performance technique from its possible side effects.

Answer D: An editor-facing interface for managing website content.

A CMS gives editors content creation, publishing, roles, and media management without editing every source file. That directly matches D's user-facing content-management interface. File storage may support it but is not its complete purpose; CSS controls appearance and JavaScript supplies client scripting. The requested role is editorial content management.

Why each choice is right or wrong

A - Incorrect: File storage can support a CMS, but does not capture its editor-facing content creation and publishing purpose.

B - Incorrect: Visual presentation is primarily CSS's role, not the defining purpose of a CMS.

C - Incorrect: Client-side scripting is a JavaScript role rather than the CMS's defining editorial function.

D - Correct: A CMS provides editors an interface and workflow for creating, publishing, and managing site content.

Primary note proof

Study-note section WEB39

> A CMS gives editors content creation, publishing, roles and media management without editing every source file.

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 9 | lines 13-21

> Web hosting is the place where all the files of your website live. Web hosting
> is a service that allows organizations and individuals to post a website or a web page
> onto the Internet. A web host, or web hosting service provider, is a business that
> provides the technologies and services needed for the website or webpage to be
> viewed in the Internet. Websites are hosted, or stored, on special computers called
> servers. When Internet users want to view your website, all they need to do is type
> your website address or domain (the domain name for the site has to be registered)
> into their browser. Their computer will then connect to your server and your webpages
> will be delivered to them through the browser.

Web Programming module | PDF page 26 | lines 4-10

> In the previous chapter we have seen how to structure a web page using HTML.
> Now it is time to add presentation to our sites, to do that we will be using a technology
> called CSS. CSS (Cascading Style Sheets) is the code that styles web content. Using
> CSS you can apply styles to make web pages look exactly how you want. This works
> because CSS is connected to the DOM(Document Object Model). CSS is not a

programming language. It's not a markup language either. CSS is a style sheet language. CSS is what you use to selectively style HTML elements.

Question sources

Computer Science MCQ practice collection · Question 63

Question 174 | AM0064

Which technique is used to improve the performance of web applications by caching frequently accessed data or resources?

A. Compression

B. Minification

C. Caching

D. Encryption

Hint

Distinguish the purpose of each development or performance technique from its possible side effects.

Answer C: Caching

WEB39 defines caching as reusing a representation or result and requiring validity or invalidation rules. Keeping frequently accessed data for reuse is therefore Caching. Compression and minification reduce transfer or source size, while encryption protects data; none of those alone stores a reusable computed result. Cache keys and privacy still matter when using the technique safely.

Why each choice is right or wrong

A - Incorrect: Compression changes transfer encoding or size rather than retaining a result for repeated reuse.

B - Incorrect: Minification removes unnecessary source characters and does not itself store frequently reused results.

C - Correct: Caching retains a representation or result so later accesses can reuse it without repeating all prior work.

D - Incorrect: Encryption protects data confidentiality rather than supplying the result-reuse technique described.

Primary note proof

Study-note section WEB39

Caching reuses a representation/result and requires validity/invalidation rules.

Study-note section WEB39

Minification removes unnecessary source characters; bundling combines files; compression changes transfer encoding; preloading fetches anticipated resources early.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 45 | lines 28-31

As a rule, only the simplest scripts are put into HTML. More complex ones reside in separate files.
The benefit of a separate file is that the browser will download it and store it in its cache.

Question sources

Computer Science MCQ practice collection · Question 64

Question 175 | AM0065

Which protocol is used for secure communication over the web?

A. HTTP

B. FTP

C. TCP/IP

D. HTTPS

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer D: HTTPS

WEB05 defines HTTPS as HTTP over TLS, protecting channel confidentiality and integrity and authenticating the server through certificate checks. That is the secure web-communication protocol among the main choices. Plain HTTP, FTP, and TCP/IP do not by their names supply that TLS-protected web exchange. The alternate option set places HTTPS at C, so its answer letter differs.

Why each choice is right or wrong

A - Incorrect: Plain HTTP does not by itself add TLS confidentiality, integrity, and certificate-based server authentication.

B - Incorrect: FTP is not the HTTPS protocol for TLS-protected web communication identified in the note.

C - Incorrect: TCP/IP names underlying networking protocols rather than the TLS-protected HTTP scheme requested.

D - Correct: HTTPS is HTTP over TLS and supplies the channel protections described in WEB05.

Primary note proof

Study-note section WEB05

HTTPS uses HTTP over TLS, protecting channel confidentiality/integrity and authenticating the server under certificate checks.

Study-note section WEB-S14

HTTP defines web application requests, response status codes and representations.

Study-note section WEB-S14

An HTTP exchange can use underlying TCP connections and IP addresses; naming that network stack does not identify the application-level method or status semantics.

Study-note section WEB-S14

FTP is a file-transfer application protocol.

Study-note section WEB-S14

SMTP transports electronic mail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 58 | lines 4-8

window.location.href returns the href (URL) of the current page
window.location.hostname returns the domain name of the web host
window.location.pathname returns the path and filename of the current page
window.location.protocol returns the web protocol used (http: or https:)
window.location.assign() loads a new document

Web Programming module | PDF page 8 | lines 29-34

• The World Wide Web (WWW) is most often called the Web.
• The Web is a network of computers all over the world.
• All the computers in the Web can communicate with each other.
• All the computers use a communication standard called HTTP.
How does the WWW work?
• Web information is stored in documents called Web pages.

Variant 1 | Alternative version 1

Which protocol is used for secure communication over the web?

A. HTTP

B. FTP

C. HTTPS

D. SMTP

Answer C: HTTPS

This variant moves HTTPS to C and replaces the fourth choice with SMTP. WEB05 still identifies HTTPS as HTTP over TLS. HTTP alone lacks that channel protection, and FTP or SMTP does not name the secure web protocol requested, so the variant answer must use its own C label.

Why each choice is right or wrong

A - Incorrect: Plain HTTP does not itself provide the TLS channel protections described in WEB05.

B - Incorrect: FTP is not the TLS-protected HTTP protocol used for the requested secure web exchange.

C - Correct: HTTPS is HTTP over TLS, protecting confidentiality, integrity, and checked server identity.

D - Incorrect: SMTP is not the HTTPS web protocol and does not answer the secure HTTP communication question.

Primary note proof

Study-note section WEB05

> HTTPS uses HTTP over TLS, protecting channel confidentiality/integrity and authenticating the server under certificate checks.

Study-note section WEB-S14

> HTTP defines web application requests, response status codes and representations.

Study-note section WEB-S14

> An HTTP exchange can use underlying TCP connections and IP addresses; naming that network stack does not identify the application-level method or status semantics.

Study-note section WEB-S14

> FTP is a file-transfer application protocol.

Study-note section WEB-S14

> SMTP transports electronic mail.

Question sources

Computer Science MCQ practice collection · Question 65

Computer Science MCQ practice collection · Question 484

Question 176 | AM0066

What is the purpose of a RESTful API in web development?

A. To control the visual appearance of web pages

B. To handle client-side scripting

C. To ensure secure data transmission over the web

D. To provide a standardized interface for accessing and manipulating web resources

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer D: A uniform interface for accessing and manipulating resources.

WEB05 describes REST constraints including client/server separation, stateless requests, and a uniform resource interface. It associates identified resources with representations and methods for reading or changing them. D expresses that purpose. CSS handles appearance, JavaScript supplies client interactivity, and HTTPS supplies channel protection; none of those is the defining role of REST itself.

Why each choice is right or wrong

A - Incorrect: Visual presentation is a CSS role, not the defining purpose of a REST resource interface.

B - Incorrect: Client-side scripting is a JavaScript role rather than the definition of REST architecture.

C - Incorrect: TLS/HTTPS protects transmission; a REST interface does not itself guarantee secure transport.

D - Correct: REST supplies a uniform resource interface for accessing representations and requesting resource operations.

Primary note proof

Study-note section WEB05

> A RESTful API follows constraints such as client/server separation, stateless requests, cacheability and a uniform resource interface.

Study-note section WEB05

> Resources have identifiers and representations; JSON is common but not required.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 21-28

> We can use JavaScript in coordination with Application programming interfaces (APIs). APIs provide us extra information. They are ready made code building blocks that are used to simplify programming in JavaScript. There are two types of APIs

- Third party API → are not built into the browser. We have to grab the code from the web. Examples: Twitter API, Google maps API etc.
- Browser API → are built into the browser. Examples DOM (Document object model) which is covered later in this chapter, Geolocation API, Canvas and WebGL, Audio and Video elements

Web Programming module | PDF page 89 | lines 20-24

6.3 Sending Request to PHP server
XMLHttpRequest (XHR) is an API that can be used by JavaScript and other web browser scripting languages to transfer and manipulate XML data to and from a webserver using HTTP, establishing an independent connection channel between a webpage's Client-Side and Server-Side.

Question sources

Computer Science MCQ practice collection · Question 66

Question 177 | AM0067

What is the purpose of responsive web design?

A. To control the visual appearance of web pages

B. To handle client-side scripting

C. To ensure secure data transmission over the web

D. To create websites that adapt and display correctly on different devices and screen sizes

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer D: To create websites that adapt and display correctly on different devices and screen sizes

The framework chapter requires support for desktop, tablet and phone screen sizes and describes responsive mobile-first websites. D directly captures this adaptation goal. Evidence: PDF p. 91.

Why each choice is right or wrong

A - Incorrect: General styling alone does not specify adaptation.

B - Incorrect: Scripting alone does not specify adaptation.

C - Incorrect: Transport security is unrelated to the described screen-size goal.

D - Correct: Exact device/screen adaptation purpose.

Primary note proof

Study-note section WEB14

Responsive design combines flexible layout, images, viewport metadata and media queries, for example @media(max-width:600px){.cards{grid-template-columns:1fr}}.

Study-note section WEB06

Meta charset declares encoding; description summarizes the page; viewport controls mobile viewport behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 91 | lines 6-13

Now a days the website you build has to support all browser types and all sizes of screen (Desktop, Tablets, Phones etc..). To make that possible we use different frameworks. This part of the chapter will introduce you to the Bootstrap framework.
Bootstrap is a free front end framework for faster and easier web development.
It is s an open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

Question sources

Computer Science MCQ practice collection · Question 67

Question 178 | AM0068

What is the purpose of version control systems (VCS) in web development?

A. To control the visual appearance of web pages

B. To handle client-side scripting

C. To ensure secure data transmission over the web

D. To track changes to source code and collaborate on web development projects

Hint

Distinguish the purpose of each development or performance technique from its possible side effects.

Answer D: Track source changes and support collaboration.

WEB39 defines VCS through recording source history, branches, and merges for review and collaboration. That directly matches D. Styling and client-side scripting are product technologies, while transport security is an HTTPS concern. A VCS can help teams review changes, but publishing a chosen build is a separate deployment action rather than its definition.

Why each choice is right or wrong

A - Incorrect: VCS records source history; CSS rather than version control determines rendered appearance.

B - Incorrect: Version control tracks scripts as files but does not execute client-side scripting.

C - Incorrect: Version history and collaboration do not themselves provide secure web transport.

D - Correct: Source history, branching, merging, and review support tracking changes and collaborating on development.

Primary note proof

Study-note section WEB39

> A VCS records source history, branches and merges to support review/collaboration; deployment publishes a chosen build and is a separate operation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 93 | lines 18-19

> 3. A suite of developer tools to help you develop, build, test, and update your code

Web Programming module | PDF page 44 | lines 13-15

> layers of web technologies are stacked together nicely, HTML will let’s structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.’

Question sources

Computer Science MCQ practice collection · Question 68

Question 179 | AM0069

Which web development concept refers to the practice of optimizing a website to rank higher in search engine results?

A. Accessibility

B. Usability

C. SEO (Search Engine Optimization)

D. Performance optimization

Hint

Distinguish the purpose of each development or performance technique from its possible side effects.

Answer C: SEO: Search Engine Optimization

WEB39 describes SEO as improving discoverability through meaningful content, headings, metadata, links, and crawlable structure. That matches the practice of improving search-result visibility. Accessibility, usability, and performance can overlap with helpful site improvements, but none is the name of this search-discoverability objective. Ranking improvement is an aim, not a guarantee from a single metadata setting.

Why each choice is right or wrong

A - Incorrect: Accessibility concerns usable access to content and controls, rather than naming the search-ranking optimization practice.

B - Incorrect: Usability concerns user experience and task completion, not specifically search-engine discoverability.

C - Correct: Search Engine Optimization names the practice of improving discoverability in search results.

D - Incorrect: Performance optimization improves resource delivery or execution speed, not the specific search-ranking practice named here.

Primary note proof

Study-note section WEB39

> SEO improves discoverability through meaningful content, headings, metadata, accessible links and crawlable structure; keyword metadata alone does not guarantee ranking.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 92 | lines 6-14

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.
> Benefits of using Node
> 1. Performance
> 2. Uses plain old JavaScript
> 3. Thousands of reusable packages
> 4. Portable

Question sources

Computer Science MCQ practice collection · Question 69

Question 180 | AM0070

Which activity is specifically used to detect XSS and SQL-injection weaknesses so they can then be repaired?

A. Encryption

B. Authentication

C. Authorization

D. Web application security testing

Hint

Identify the data source or security responsibility precisely, and check whether a combined source or a different control could also qualify.

Answer D: Web application security testing

Web application security testing can detect SQL injection and XSS and provide evidence for repairs. Detection does not by itself remove the flaws: safe query parameterization and context-appropriate output handling are still required. Encryption, authentication and authorization address other protection goals and do not perform the vulnerability-testing activity requested. Changing the stem from a claim of prevention to a detection task preserves the security-testing concept without overstating its effect.

Why each choice is right or wrong

A - Incorrect: Encryption protects content but is not the requested detection activity.

B - Incorrect: Authentication checks identity rather than testing for these code weaknesses.

C - Incorrect: Authorization checks permitted actions rather than conducting vulnerability tests.

D - Correct: This is the testing activity used to discover the specified weaknesses.

Primary note proof

Study-note section WEB27

> A security test can detect SQL injection or XSS but does not itself fix either.

Study-note section WEB27

> Bind SQL parameters to keep values separate from commands.

Study-note section WEB-S11

> Authentication verifies a claimed identity.

Study-note section WEB-S11

> Authorization checks whether a particular entity is permitted to perform a requested action.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 73 | lines 2-10

Form validation is done to ensure that the user has provided the relevant information. Basic validation can be done using HTML elements. For example, in the above script, the email address text box is having a type value as “email”, which prevents the user from entering the incorrect value for an email. Every form field in the above script is followed by a required attribute, which will intimate the user not to leave any field empty before submitting the form. PHP methods and arrays used in form processing are:
• isset(): This function is used to determine whether the variable or a form control is having a value or not.

Web Programming module | PDF page 80 | lines 2-15

5.13.3 Sending Query to MySQL Database using PHP
A database query from PHP is basically a MySQL command wrapped up in a tiny PHP function called mysql_query(). This is where you use the basic SQL workhorses of SELECT,
INSERT, UPDATE, and DELETE, CREATE or DROP a table (but not those to create or drop an entire database) can be used with this PHP function.
Syntax:
[$resultvar=] mysql_query(“query”,[link identifier]);
$resultvar - is an optional result variable. This is used when the query is a select statement.
Query - is any valid insert, delete, update, select, create table or drop table statement enclosed by quotes. It is a good habit to hold the query in a separate variable to add flexibility to your code.
Link identifier-is an optional link identifier

Question sources

Computer Science MCQ practice collection · Question 70

Question 181 | AM0071

A server request handler contains this fragment, with no user-defined syntax extensions:
$visitor = "Mimi";
echo "Hello, " . $visitor;
Which listed language accepts this fragment as written in its normal language environment?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Trace the bindings and operations in order; preserve the exact string characters and distinguish returned values from emitted output.

Answer D: PHP

The fragment uses a dollar-prefixed variable, the echo output construct and dot concatenation. Together these are valid PHP syntax, producing the greeting by joining two strings. JavaScript uses different output calls and does not use dot as this binary concatenation operator. HTML and CSS provide structure and presentation rather than directly executing this server-side statement sequence. Recognizing the actual syntax resolves the language choice without excluding JavaScript from server programming generally.

Why each choice is right or wrong

A - Incorrect: HTML markup does not execute this variable assignment and output-construct sequence.

B - Incorrect: CSS presentation rules do not execute this statement sequence.

C - Incorrect: The echo construct and dot concatenation shown are not ordinary JavaScript syntax.

D - Correct: PHP accepts the assignment, echo construct and dot concatenation as written.

Primary note proof

Study-note section WEB23

> Variables begin with $, then a letter/underscore and subsequent letters/digits/underscores; $AGE18 works but $18AGE does not.

Study-note section WEB24

> echo is a language construct with no return value and can emit comma-separated expressions.

Study-note section WEB24

> Dot concatenates; + is arithmetic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 4-10

> PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Web Programming module | PDF page 92 | lines 6-9

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.

Web Programming module | PDF page 10 | lines 9-16

> HTML (HyperText Markup Language) is the most basic building block of the Web.
> HTML is not a programming language rather it is a markup language that tells web browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
> HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Question sources

Computer Science MCQ practice collection · Question 71

Question 182 | AM0072

Which server-side scripting language is often used for building scalable and high-performance web applications?

A. HTML

B. CSS

C. JavaScript

D. Node.js

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer C: JavaScript, executed server-side by runtimes such as Node.js.

WEB01 directly distinguishes JavaScript as a language and Node.js as its runtime. The question asks for the language, so C is the category-correct option. WEB36 warns that long CPU work or synchronous I/O can block Node callbacks, so scalability is architectural and workload-dependent rather than a guarantee from the language name alone.

Why each choice is right or wrong

A - Incorrect: HTML is markup for structure rather than server-side application scripting.

B - Incorrect: CSS is presentation rather than the server application language requested.

C - Correct: JavaScript is the language that runs in Node-based server applications.

D - Incorrect: Node.js is a runtime, not a distinct server-side scripting language.

Primary note proof

Study-note section WEB01

> Node.js is a JavaScript runtime, not a separate language.

Study-note section WEB36

One event-loop thread can handle many waiting requests, but long CPU loops or synchronous I/O block its callbacks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 92 | lines 6-14

b. Node.js
Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.
Benefits of using Node
1. Performance
2. Uses plain old JavaScript
3. Thousands of reusable packages
4. Portable

Web Programming module | PDF page 10 | lines 9-16

HTML (HyperText Markup Language) is the most basic building block of the Web. HTML is not a programming language rather it is a markup language that tells web browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Question sources

Computer Science MCQ practice collection · Question 72

Question 183 | AM0073

What is the purpose of server-side scripting languages in web development?

A. To control the visual appearance of web pages

B. To handle client-side interactivity

C. To generate dynamic content and interact with databases

D. To ensure secure data transmission over the web

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: To generate dynamic content and interact with databases

Server PHP generates dynamic output and connects sites to backend databases, enabling two-way communication (p.60). C combines those explicitly described purposes. Evidence: PDF p. 60, PDF p. 44.

Why each choice is right or wrong

A - Incorrect: CSS presentation role.

B - Incorrect: Client JavaScript role.

C - Correct: Exact dynamic-output/database role.

D - Incorrect: Transport security is not the stated server scripting purpose.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 9-19

> language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.
> Client-side scripting is the glamorous, eye-catching part of Web development. In contrast, server-side scripting is invisible to the user. Server-side Web scripting is mostly about connecting Web sites to back-end servers, such as databases. This enables the following types of two-way communication:
> ➢ Server to client: Web pages can be assembled from back end-server output.
> ➢ Client to server: Customer-entered information can be acted upon.
> Common examples of client-to-server interaction are online forms with some drop-down lists (usually the ones that require you to click a button) that the script assembles dynamically on the server.

Web Programming module | PDF page 44 | lines 13-15

> layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'

Question sources

Computer Science MCQ practice collection · Question 73

Question 184 | AM0074

Which of the following is a server-side scripting language for Microsoft's ASP.NET framework?

A. Python

B. Ruby

C. C#

D. Java

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: C#

WEB01 explicitly states that C# is commonly used with ASP.NET. That identifies the requested language among the main options. Python and Java can be used for server applications in other roles described here, but are not its stated ASP.NET pairing. The answer concerns language/framework association, not a claim that only one language can ever interact with the platform.

Why each choice is right or wrong

A - Incorrect: Python is a server-side language, but it is not the ASP.NET language pairing identified in WEB01.

B - Incorrect: Ruby is not the language explicitly paired with ASP.NET in the supplied note.

C - Correct: WEB01 explicitly states that C# is commonly used with ASP.NET.

D - Incorrect: The note associates Java with JSP/Java server applications, while identifying C# for ASP.NET.

Primary note proof

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 3-10

> 5.1 Introduction to PHP
> PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the

language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science MCQ practice collection · Question 74

Question 185 | AM0075

Which client-side scripting language is supported by all modern web browsers?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: JavaScript

The JavaScript chapter describes browser functionality and modern browser JavaScript interactivity. C matches the client scripting language; HTML and CSS have different structural/presentation roles and PHP is server-side. Evidence: PDF p. 44, PDF p. 56, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: Markup language, not scripting.

B - Incorrect: Stylesheet language, not scripting.

C - Correct: Browser scripting language described throughout chapter 4.

D - Incorrect: Explicit server-side language.

Primary note proof

Study-note section WEB01

HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 9-15

> JavaScript is a scripting or programming language that allows us to implement complex features on web pages. JavaScript will enable us to display timely content updates, interactive maps, 2D/3D animated graphics etc. JavaScript is one of the three layers of web technologies which lets to add functionality to a web page. The three layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'

Web Programming module | PDF page 56 | lines 11-16

> There are no official standards for the Browser Object Model (BOM).
> Since modern browsers have implemented (almost) the same methods and properties for JavaScript interactivity, it is often referred to, as methods and properties of the BOM.
> The window object is supported by all browsers. It represents the browser's window.

Web Programming module | PDF page 60 | lines 4-10

> PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science MCQ practice collection · Question 75

Question 186 | AM0076

What is the purpose of client-side scripting languages in web development?

A. To control the visual appearance of web pages

B. To handle server-side processing and database interactions

C. To generate dynamic content and interact with databases

D. To add interactivity and dynamic behavior to web pages

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: To add interactivity and dynamic behavior to web pages

JavaScript implements timely updates, interactive maps and animated graphics (p.44). This is the client-side interactivity/dynamic-behavior purpose in D. Evidence: PDF p. 44, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: CSS presentation is a different layer.

B - Incorrect: Server/database processing belongs to server scripting.

C - Incorrect: Server-generated database output is a different role.

D - Correct: Matches interactive client behavior.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 9-19

> JavaScript is a scripting or programming language that allows us to implement complex features on web pages. JavaScript will enable us to display timely content updates, interactive maps, 2D/3D animated graphics etc. JavaScript is one of the three layers of web technologies which lets to add functionality to a web page. The three layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'
> What can JavaScript really do? It can
> - Stores values inside variables
> - Perform operations on pieces of texts
> - Run a code in response to events

Web Programming module | PDF page 60 | lines 11-16

> Client-side scripting is the glamorous, eye-catching part of Web development. In contrast, server-side scripting is invisible to the user. Server-side Web scripting is mostly about connecting Web sites to back-end servers, such as databases. This enables the following types of two-way communication:
> ➢ Server to client: Web pages can be assembled from back end-server output.
> ➢ Client to server: Customer-entered information can be acted upon.

Question sources

Computer Science MCQ practice collection · Question 76

Question 187 | AM0077

Which client-side scripting language is primarily used for manipulating the Document Object Model (DOM) of a web page?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Track which node is selected or created, and distinguish changing content from attaching a node.

Answer C: JavaScript

The module JavaScript DOM section explains accessing elements and changing document structure, style and content. That makes JavaScript the client scripting language for DOM manipulation. Evidence: PDF p. 52, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: HTML supplies structure rather than the client scripting operation.

B - Incorrect: CSS styles content rather than being the scripting language.

C - Correct: Explicitly uses DOM APIs to access/change pages.

D - Incorrect: PHP executes on the server in this module.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB20

> The DOM is a hierarchy of nodes representing the document.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 52 | lines 3-20

> 4.10 JavaScript DOM (Document object Model)
> The Document Object Model (DOM) is the data representation of the objects that comprise the structure and content of a document on the web. The Document Object Model (DOM) is a programming interface for web documents. It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects; that way, programming languages can interact with the page.
> 4.10.1 Accessing HTML elements in JavaScript

In the DOM Api, there are different methods to get access to HTML elements. We will list the available methods here, but only the most recent methods are covered here.
- getElementById(id) returns an element with that specific ID
- getElmentsByClassName(class) returns elements with that specific class
getElementsByTagName(tag) returns elements with the given tag name
- querySelector(selector) → The Document method querySelector() returns the first Element within the document that matches the specified selector, or group of selectors. If no matches are found, null is returned.
element = document.querySelector(selectors);

Web Programming module | PDF page 60 | lines 4-10

PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science MCQ practice collection · Question 77

Question 188 | AM0078

Which client-side scripting language is often used for validating form inputs on web pages?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Separate event dispatch, handler registration and cancellation of the browser default action.

Answer C: JavaScript

The module includes a JavaScript form-validation example checking an empty field and prompting the user. C identifies the scripting language performing that check. Evidence: PDF p. 55, PDF p. 73.

Why each choice is right or wrong

A - Incorrect: HTML can provide basic validation but is not the requested scripting language.

B - Incorrect: Presentation language, not this validation script.

C - Correct: Exact language of the documented validation example.

D - Incorrect: PHP validates on the server, not in the browser.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB21

> For submission, form.addEventListener("submit",e=>{if(!valid()){e.preventDefault();}}) stops the default only on failure.

Study-note section WEB21

> Browser checks improve feedback; repeat validation and authorization on the server.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 55 | lines 4-18

> Now inside the submit event handler function we call the event.preventDefault()
> method to prevent the default behavior and avoid a form submit to reload the page,
> this gives us control. After we gain control over the submit event we can all kind of
> stuff, like form validation
> Here is a simple but different way of validating a form using javascript
> Form Validation
> <html>
> <head>
> <title>Form Validation</title>
> <script type = "text/javascript">
> if(document.myForm.Name.value == "") {
> alert("Please provide your name!");
> document.myForm.Name.focus() ;
> return false;
> }

Web Programming module | PDF page 73 | lines 2-7

> Form validation is done to ensure that the user has provided the relevant information.
> Basic validation can be done using HTML elements. For example, in the above
> script, the email address text box is having a type value as “email”, which prevents
> the user from entering the incorrect value for an email. Every form field in the above
> script is followed by a required attribute, which will intimate the user not to leave
> any field empty before submitting the form. PHP methods and arrays used in form

Question sources

Computer Science MCQ practice collection · Question 78

Question 189 | AM0079

Which client-side scripting technique allows for loading new content into a web page without a full page reload?

A. AJAX

B. JSON

C. XML

D. Bootstrap

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer A: AJAX

AJAX explicitly requests data and updates a page without full refresh. JSON/XML are possible exchange formats, and Bootstrap is a responsive UI framework. Evidence: PDF p. 89, PDF p. 91.

Why each choice is right or wrong

A - Correct: Exact no-refresh exchange technique.

B - Incorrect: A supported data format, not the exchange technique.

C - Incorrect: Another data format, not the technique.

D - Incorrect: Responsive framework, not the AJAX request mechanism.

Primary note proof

Study-note section WEB34

> AJAX means Asynchronous JavaScript and XML, but responses can be JSON, text, HTML or XML.

Study-note section WEB38

> Bootstrap supplies layout/components.

Study-note section WEB35

> JSON represents objects, arrays, strings, numbers, Booleans and null; keys/strings use double quotes, and functions/undefined are not JSON values.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 4-18

> AJAX is a web development technique for creating interactive web applications.
> AJAX stands for Asynchronous JavaScript and XML. It can send and receive
> information in various formats, including JSON, XML, HTML, and text files. AJAX

is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script. AJAX's most appealing characteristic is its "asynchronous" nature, which means it can communicate with the server, exchange data, and update the page without having to refresh the page.
The two major features of AJAX allow you to do the following:
- Make requests to the server without reloading the page
- Receive and work with data from the server
6.2 XMLHttpRequest Object
XMLHttpRequest (XHR) objects are used to interact with servers. You can retrieve data from a URL without having to do a full-page refresh. This enables a Web page to update just part of a page without disrupting what the user is doing. Despite its name, XMLHttpRequest can be used to retrieve any type of data, not just XML.

Web Programming module | PDF page 91 | lines 10-13

Bootstrap is a free front end framework for faster and easier web development. It is s an open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

Question sources

Computer Science MCQ practice collection · Question 79

Question 190 | AM0080

A course-catalog page already has CSS that switches between one and three columns at the required viewport widths. It still needs a browser-side filter that updates the visible course cards without a full-page reload. With suitable application code, which listed technologies can implement that interaction while retaining the CSS layout? Select all that apply.

A. jQuery

B. React

C. Angular

D. Vue.js

Hint

Match the stated application requirement to the capabilities and category of each technology.

Answer A, B, C, D: jQuery; React; Angular; Vue.js

The responsive layout is already supplied by CSS; the remaining task is programmable interface behavior. jQuery can update the DOM, and React, Angular and Vue can implement the filter and visible-card updates through their interface facilities. None is uniquely required by this interaction. All four remain possible choices when suitable application code is supplied, without implying that selecting any toolkit automatically implements responsiveness or the filter itself.

Why each choice is right or wrong

A - Correct: jQuery can handle filter interaction and update existing DOM content while the CSS controls layout.

B - Correct: React can render the filtered card interface while using the supplied responsive CSS.

C - Correct: Angular can implement the filter interaction and visible-card updates with the existing layout styles.

D - Correct: Vue can update visible cards in response to filter input while retaining responsive styles.

Primary note proof

Study-note section WEB38

> jQuery simplifies DOM operations; React builds component interfaces; Angular is a component platform, distinct from older AngularJS; Vue also builds interactive interfaces.

Study-note section WEB38

> Each can contribute to a responsive interface, so those generic descriptions do not identify one unique framework.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 94 | lines 14-17

> d. React.js
> React.js is an open-source front-end JavaScript library for building user interfaces based on UI components. React builds encapsulated component that manage their own state, then compose them to complex UIs. To build for the web, developers use

Web Programming module | PDF page 93 | lines 12-19

> c. Angular.js
> Angular is a development platform, built on TypeScript. As a platform, Angular includes:
> 1. A component-based framework for building scalable web applications
> 2. A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more
> 3. A suite of developer tools to help you develop, build, test, and update your code

Web Programming module | PDF page 91 | lines 28-31

> 2. Many of Bootstrap components require the use of JavaScript to function.
> Specifically, they require jQuery, Popper.js, and boostrap JavaScript plugins.
> Follow this order when including the JS files
> - JQuery

Question sources

Computer Science MCQ practice collection · Question 80

Question 191 | AM0081

A page sends asynchronous HTTP requests and uses each response to update part of the page without a full navigation. Which listed technique names this browser request-and-update pattern?

A. AJAX

B. JSON

C. Websockets

D. RESTful API

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer A: AJAX

AJAX is the technique of using asynchronous browser HTTP requests to obtain data and update part of a page. It can carry JSON, XML or other response formats. JSON names a data representation, while WebSocket supplies a persistent bidirectional message channel rather than the stated repeated HTTP request/response pattern. A RESTful API describes the server interface organization, not the browser update technique.

Why each choice is right or wrong

A - Correct: AJAX names the asynchronous HTTP request-and-partial-update technique described.

B - Incorrect: JSON can encode a response but does not perform browser requests or update the page.

C - Incorrect: WebSocket uses a persistent bidirectional channel; that is not the specified HTTP request/response technique.

D - Incorrect: A RESTful API can serve the requests, but names an interface style rather than the client’s request-and-update technique.

Primary note proof

Study-note section WEB34

> AJAX means Asynchronous JavaScript and XML, but responses can be JSON, text, HTML or XML.

Study-note section WEB34

> Configure an XHR with open("GET",url,true), register response/error handlers and call send().

Study-note section WEB35

WebSocket establishes a persistent bidirectional message channel after its handshake.

Study-note section WEB35

JSON represents objects, arrays, strings, numbers, Booleans and null; keys/strings use double quotes, and functions/undefined are not JSON values.

Study-note section WEB05

A RESTful API follows constraints such as client/server separation, stateless requests, cacheability and a uniform resource interface.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 4-18

AJAX is a web development technique for creating interactive web applications.
AJAX stands for Asynchronous JavaScript and XML. It can send and receive information in various formats, including JSON, XML, HTML, and text files. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script. AJAX's most appealing characteristic is its "asynchronous" nature, which means it can communicate with the server, exchange data, and update the page without having to refresh the page.
The two major features of AJAX allow you to do the following:
- Make requests to the server without reloading the page
- Receive and work with data from the server
6.2 XMLHttpRequest Object
XMLHttpRequest (XHR) objects are used to interact with servers. You can retrieve data from a URL without having to do a full-page refresh. This enables a Web page to update just part of a page without disrupting what the user is doing. Despite its name, XMLHttpRequest can be used to retrieve any type of data, not just XML.

Question sources

Computer Science MCQ practice collection · Question 81

Question 192 | AM0082

Which techniques store data locally in a user's browser? Select all that apply.

A. Cookies

B. Local storage

C. Session storage

D. IndexedDB

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A, B, C, D: Cookies; Local storage; Session storage; IndexedDB

All four entries are browser storage techniques, although their persistence, data model and automatic transmission behavior differ. Cookies hold browser name/value data that can accompany matching requests. localStorage persists origin-specific strings, sessionStorage is scoped to a page session or tab, and IndexedDB supports structured local data. The revised question asks for every local-storage technique rather than assuming that only the API named local storage qualifies.

Why each choice is right or wrong

A - Correct: Cookies are browser-held name/value data, even though matching requests can transmit them.

B - Correct: localStorage persists origin-specific string values until removed or cleared.

C - Correct: sessionStorage stores browser data within a page-session or tab scope.

D - Correct: IndexedDB supplies structured local browser storage.

Primary note proof

Study-note section WEB28

> Cookies are browser-held name/value data automatically attached to matching requests subject to domain, path, expiry, Secure and SameSite rules.

Study-note section WEB28

> localStorage persists origin-specific string values across sessions until removed/cleared; sessionStorage is scoped to a page session/tab.

Study-note section WEB28

> IndexedDB supplies structured local storage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 58 | lines 12-23

> Cookies let you store user information in web pages.
> What are Cookies?
> Cookies are data, stored in small text files, on your computer.
> When a web server has sent a web page to a browser, the connection is shut down,
> and the server forgets everything about the user. Cookies were invented to solve the
> problem "how to remember information about the user":
> When a user visits a web page, his/her name can be stored in a cookie.
> Next time the user visits the page, the cookie "remembers" his/her name.
> Cookies are saved in name-value pairs like:
> When a browser requests a web page from a server, cookies belonging to the page
> are added to the request. This way the server gets the necessary data to "remember"
> information about users.

Web Programming module | PDF page 76 | lines 2-5

> in one application. A PHP session allows you to store user information on the server for later use (i.e., username, shopping items, etc.). However, session information is temporary and will be deleted after the user has left the website. If you need a permanent storage you may want to store the data in a database.

Question sources

Computer Science MCQ practice collection · Question 82

Question 193 | AM0083

Which source-processing step removes unnecessary characters from CSS or JavaScript, distinct from bundling files together or compressing transfer encoding?

A. Compression

B. Caching

C. Minification

D. Encryption

Hint

Distinguish the purpose of each development or performance technique from its possible side effects.

Answer C: Minification

Minification removes unnecessary source characters, such as dispensable formatting, while preserving the intended program or stylesheet behavior. Bundling combines files and compression changes transfer encoding; these are separate operations even when a build pipeline performs them together. Caching instead reuses an existing representation, and encryption serves confidentiality. Restricting the question to source-character reduction makes minification the supported technique without claiming it also performs file combination.

Why each choice is right or wrong

A - Incorrect: Compression changes transfer encoding rather than performing the specified source-character reduction.

B - Incorrect: Caching reuses a representation rather than rewriting its source characters.

C - Correct: Minification is the source-character reduction step.

D - Incorrect: Encryption protects confidentiality rather than simplifying source formatting.

Primary note proof

Study-note section WEB39

> Minification removes unnecessary source characters; bundling combines files; compression changes transfer encoding; preloading fetches anticipated resources early.

Study-note section WEB39

> Caching reuses a representation/result and requires validity/invalidation rules.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 45 | lines 28-31

> As a rule, only the simplest scripts are put into HTML. More complex ones reside in separate files.
> The benefit of a separate file is that the browser will download it and store it in its cache.

Question sources

Computer Science MCQ practice collection · Question 83

Question 194 | AM0084

Which client-side technique is used to handle and process user interactions, such as mouse clicks or keyboard inputs?

A. Event handling

B. Form validation

C. Responsive design

D. Browser compatibility

Hint

Separate event dispatch, handler registration and cancellation of the browser default action.

Answer A: Event handling

Events signal user actions and handlers run functions in response, including click events (p.53). A names this mechanism. Validation is a particular task, and responsive design/browser compatibility do not name action dispatch. Evidence: PDF p. 53.

Why each choice is right or wrong

A - Correct: Exact handler mechanism.

B - Incorrect: Checks fields rather than the general interaction mechanism.

C - Incorrect: Adapts layout rather than dispatching user actions.

D - Incorrect: Cross-browser support is not the event dispatch mechanism.

Primary note proof

Study-note section WEB21

> An event handler runs when an event occurs: click, input, keydown, mouseover, submit and load are examples.

Study-note section WEB14

Browser compatibility requires feature testing and fallbacks; responsive size handling is a different issue.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 53 | lines 4-18

4.10.3 Events in JavaScript
An event is a signal that something has happened. All DOM nodes generate such signals (but events are not limited to DOM).
Here's a list of the most useful DOM events, just to take a look at:
- click - when the mouse clicks on an element (touchscreen devices generate it on a tap).
- contextmenu - when the mouse right-clicks on an element.
- mouseover / mouseout - when the mouse cursor comes over / leaves an element.
- mousedown / mouseup - when the mouse button is pressed / released over an element.
- mousemove - when the mouse is moved.
Event Handlers
To react on events we can assign a handler - a function that runs in case of an event.
Handlers are a way to run JavaScript code in case of user actions.

Question sources

Computer Science MCQ practice collection · Question 84

Question 195 | AM0085

Which client-side technique is used to ensure that a web page displays correctly across different devices and screen sizes?

A. Event handling

B. Form validation

C. Responsive design

D. Browser compatibility

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer C: Responsive design

Responsive websites support desktop/tablet/phone screen sizes in the Bootstrap chapter. That fits devices and screen sizes; browser compatibility is a related but different browser-engine concern. Evidence: PDF p. 91.

Why each choice is right or wrong

A - Incorrect: Handles actions rather than screen size.

B - Incorrect: Checks input rather than screen size.

C - Correct: Exact screen adaptation concept.

D - Incorrect: Different browsers are a separate axis from device dimensions.

Primary note proof

Study-note section WEB14

> Responsive design combines flexible layout, images, viewport metadata and media queries, for example @media(max-width:600px){.cards{grid-template-columns:1fr}}.

Study-note section WEB14

> Browser compatibility requires feature testing and fallbacks; responsive size handling is a different issue.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 91 | lines 6-13

> Now a days the website you build has to support all browser types and all sizes of screen (Desktop, Tablets, Phones etc..). To make that possible we use different frameworks. This part of the chapter will introduce you to the Bootstrap framework.
> Bootstrap is a free front end framework for faster and easier web development.
> It is s an open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

Question sources

Computer Science MCQ practice collection · Question 85

Question 196 | AM0086

Which browser technique sends an asynchronous HTTP request, then processes a JSON response through a callback or promise without reloading the whole page? Choose the technique, rather than a resource-interface style or a persistent messaging protocol.

A. AJAX

B. Websockets

C. RESTful API

D. Bootstrap

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer A: AJAX

AJAX describes asynchronous browser/server requests and response handling; despite its name, the returned data need not be XML and can be JSON. The scenario specifically uses an HTTP request with a later handler rather than a persistent bidirectional messaging channel. A RESTful API describes a resource-interface architecture that the request might use, while Bootstrap supplies interface components. Those can coexist with the interaction but do not name the requested browser technique.

Why each choice is right or wrong

A - Correct: AJAX names the asynchronous HTTP request-and-response technique described.

B - Incorrect: WebSocket is a persistent bidirectional messaging channel rather than this HTTP-request technique.

C - Incorrect: REST describes resource-interface constraints, not the requested browser execution technique.

D - Incorrect: Bootstrap supplies interface components, not this request-handling technique.

Primary note proof

Study-note section WEB34

> AJAX means Asynchronous JavaScript and XML, but responses can be JSON, text, HTML or XML.

Study-note section WEB34

> A callback is a function invoked later by the runtime/API.

Study-note section WEB35

> WebSocket establishes a persistent bidirectional message channel after its handshake.

Study-note section WEB05

> A RESTful API follows constraints such as client/server separation, stateless requests, cacheability and a uniform resource interface.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 4-13

> AJAX is a web development technique for creating interactive web applications.
> AJAX stands for Asynchronous JavaScript and XML. It can send and receive
> information in various formats, including JSON, XML, HTML, and text files. AJAX
> is a new technique for creating better, faster, and more interactive web applications
> with the help of XML, HTML, CSS, and Java Script. AJAX’s most appealing
> characteristic is its "asynchronous" nature, which means it can communicate with the

server, exchange data, and update the page without having to refresh the page.
The two major features of AJAX allow you to do the following:
- Make requests to the server without reloading the page
- Receive and work with data from the server

Web Programming module | PDF page 91 | lines 10-13

Bootstrap is a free front end framework for faster and easier web development.
It is s an open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

Question sources

Computer Science MCQ practice collection · Question 86

Question 197 | AM0087

Which listed technologies can implement interactive behaviors such as tooltips and modals, with suitable application code or components? Select all that apply.

A. jQuery

B. React

C. Angular

D. Vue.js

Hint

Match the stated application requirement to the capabilities and category of each technology.

Answer A, B, C, D: jQuery; React; Angular; Vue.js

Tooltips and modals are interface behaviors, not a unique signature of one library or platform. jQuery can manipulate the DOM, while React, Angular and Vue can implement such interactions through component-based application logic. The question explicitly permits suitable code or components, so it does not claim every behavior is bundled into each technology by default. All four can contribute and therefore belong in the selection.

Why each choice is right or wrong

A - Correct: jQuery DOM operations can implement the stated interactions.

B - Correct: React components can implement the stated interactions.

C - Correct: Angular components can implement the stated interactions.

D - Correct: Vue components can implement the stated interactions.

Primary note proof

Study-note section WEB38

jQuery simplifies DOM operations; React builds component interfaces; Angular is a component platform, distinct from older AngularJS; Vue also builds interactive interfaces.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 91 | lines 19-29

3. Has lots of component and a good grid system
4. It creates Platform-independent web pages.
5. It creates Responsive Web-pages.
6. It is designed to be responsive to mobile devices too.
How to use Bootstrap
To add bootstrap you can either use the bootstrapCDN or you can download it and use it locally. Here are the steps to add the bootstrapCDN to our project,
1. Copy-paste the stylesheet <link> into your <head> before all other stylesheets to load our CSS.
2. Many of Bootstrap components require the use of JavaScript to function. Specifically, they require jQuery, Popper.js, and boostrap JavaScript plugins.

Web Programming module | PDF page 94 | lines 14-17

d. React.js
React.js is an open-source front-end JavaScript library for building user interfaces based on UI components. React builds encapsulated component that manage their own state, then compose them to complex UIs. To build for the web, developers use

Web Programming module | PDF page 93 | lines 12-19

c. Angular.js
Angular is a development platform, built on TypeScript. As a platform, Angular includes:
1. A component-based framework for building scalable web applications
2. A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more
3. A suite of developer tools to help you develop, build, test, and update your code

Question sources

Computer Science MCQ practice collection · Question 87

Question 198 | AM0088

Which client-side technique is used to create visually appealing and responsive layouts for web pages?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer B: CSS

CSS is explicitly presentation code, including layout properties; HTML supplies structure, JavaScript behavior and PHP server processing. B is the intended layout-styling technology. Evidence: PDF p. 26, PDF p. 37, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: Structure rather than the principal styling layer.

B - Correct: Styles and layout match the CSS presentation role.

C - Incorrect: Behavior rather than primary layout styling.

D - Incorrect: Server output rather than browser styling rules.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB14

> Responsive design combines flexible layout, images, viewport metadata and media queries, for example @media(max-width:600px){.cards{grid-template-columns:1fr}}.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 26 | lines 4-16

> In the previous chapter we have seen how to structure a web page using HTML.
> Now it is time to add presentation to our sites, to do that we will be using a technology called CSS. CSS (Cascading Style Sheets) is the code that styles web content. Using CSS you can apply styles to make web pages look exactly how you want. This works because CSS is connected to the DOM(Document Object Model). CSS is not a programming language. It's not a markup language either. CSS is a style sheet

language. CSS is what you use to selectively style HTML elements.
p {
color: red;
}
3.1.1 Introduction to CSS
CSS stands for Cascading Style Sheet. It is a stylesheet language used to describe the presentation of a document written in HTML. Styles define how to display HTML

Web Programming module | PDF page 37 | lines 15-18

Display
The display CSS property sets whether an element is treated as a block or inline element and the layout used for its children, such as flow layout, grid or flex. Formally, the display property sets an element's inner and outer display types. The

Web Programming module | PDF page 60 | lines 4-10

PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science MCQ practice collection · Question 88

Question 199 | AM0089

Which client-side technique is used to optimize web page loading speed by preloading resources in advance?

A. Compression

B. Minification

C. Caching

D. Preloading

Hint

Distinguish the purpose of each development or performance technique from its possible side effects.

Answer D: Preloading

WEB39 defines preloading as fetching anticipated resources early. That exactly matches preloading resources in advance in the stem. Compression and minification change transfer or source size, while caching reuses a representation already obtained. Preloading can help loading speed when chosen well, but excessive early fetching can compete with critical work, so performance should be measured.

Why each choice is right or wrong

A - Incorrect: Compression changes transfer representation or size rather than initiating anticipated resource fetches early.

B - Incorrect: Minification removes unnecessary source characters rather than prefetching resources before demand.

C - Incorrect: Caching reuses retained results; it is distinct from initiating an early fetch of anticipated resources.

D - Correct: Preloading fetches anticipated resources early, matching the advance-loading technique described.

Primary note proof

Study-note section WEB39

> Minification removes unnecessary source characters; bundling combines files; compression changes transfer encoding; preloading fetches anticipated resources early.

Study-note section WEB39

> Measure performance because excess preloading/bundling may compete with critical work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 45 | lines 28-31

> As a rule, only the simplest scripts are put into HTML. More complex ones reside
> in separate files.
> The benefit of a separate file is that the browser will download it and store it in its
> cache.

Question sources

Computer Science MCQ practice collection · Question 89

Question 200 | AM0090

Which client-side technique is used to ensure that a web page displays correctly in different web browsers?

A. Event handling

B. Form validation

C. Responsive design

D. Browser compatibility

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer D: Browser compatibility

The framework chapter says sites must support all browser types as well as screen sizes. Different-browser correctness is browser compatibility; responsive design concerns the separate size/device aspect. Evidence: PDF p. 91.

Why each choice is right or wrong

A - Incorrect: Handles interactions rather than browser differences.

B - Incorrect: Checks form input rather than browser rendering support.

C - Incorrect: Screen-size adaptation is a separate requirement.

D - Correct: Names the cross-browser correctness requirement.

Primary note proof

Study-note section WEB14

> Browser compatibility requires feature testing and fallbacks; responsive size handling is a different issue.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 91 | lines 6-9

> Now a days the website you build has to support all browser types and all sizes of screen (Desktop, Tablets, Phones etc..). To make that possible we use different frameworks. This part of the chapter will introduce you to the Bootstrap framework.

Question sources

Computer Science MCQ practice collection · Question 90

Question 201 | AM0091

What is the purpose of HTML in web programming?

A. To control the visual appearance of web pages

B. To handle server-side processing and database interactions

C. To specify the structure and content of web pages

D. To add interactivity and dynamic behavior to web pages

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: To specify the structure and content of web pages

HTML defines the meaning and structure of web content. The module separately assigns appearance to CSS and behavior to JavaScript, while PHP handles server-side output. Evidence: PDF p. 10, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: CSS presentation role.

B - Incorrect: Server scripting/database role.

C - Correct: Exact HTML structure/content role.

D - Incorrect: JavaScript behavior role.

Primary note proof

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 10 | lines 9-16

> HTML (HyperText Markup Language) is the most basic building block of the Web.
> HTML is not a programming language rather it is a markup language that tells web browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
> HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Web Programming module | PDF page 60 | lines 11-16

Client-side scripting is the glamorous, eye-catching part of Web development. In contrast, server-side scripting is invisible to the user. Server-side Web scripting is mostly about connecting Web sites to back-end servers, such as databases. This enables the following types of two-way communication:
➢ Server to client: Web pages can be assembled from back end-server output.
➢ Client to server: Customer-entered information can be acted upon.

Question sources

Computer Science MCQ practice collection · Question 91

Question 202 | AM0092

Which HTML element is used to create a heading in a web page?

A. <p>

B.

C. <h1>

D. <a>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C: <h1>

Heading formats range from h1 to h6 (p.15). The displayed h1 choice is one of these; p is paragraph, a is link, and div is not identified as a heading format. Evidence: PDF p. 15, PDF p. 15, PDF p. 14.

Why each choice is right or wrong

A - Incorrect: Paragraph rather than heading.

B - Incorrect: Not one of the documented heading tags.

C - Correct: Documented heading tag.

D - Incorrect: Anchor/link rather than heading.

Primary note proof

Study-note section WEB06

Heading levels h1-h6 express hierarchy, with h6 conventionally smallest in default styling, not a mandatory CSS size.

Study-note section WEB06

<p> marks paragraphs;
 is a line break and <hr> a thematic break.

Study-note section WEB07

Use Read for a link; href is the destination, not url/link.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 15 | lines 5-12

Headings -
Headings are predefined formats for text presentation and there are six heading formats defined in HTML:
<h1> up to <h6>
<h1> the largest font size
<h6> the smallest font size
Format :<h1>...</h1>
E.g., <h2>a text in heading two</h2>

Web Programming module | PDF page 15 | lines 27-30

Paragraph: definition of paragraph
Format: ...
E.g., this is a paragraph of text. it has a new line before and after it.
The browser inserts a new line before and after the text in the paragraph tag.

Web Programming module | PDF page 14 | lines 5-9

In HTML, a link is created by wrapping the text or other content inside an <a> tag (anchor tag) and using the href attribute, also known as Hypertext reference, or target that contains the web address.
Here is an example of a link to the google website
Google.

Question sources

Computer Science MCQ practice collection · Question 92

Question 203 | AM0093

Which HTML element is used to create a paragraph in a web page?

A. <h1>

B.

C.

D. <a>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C:

The paragraph section gives ... explicitly. Headings and anchors have their own different formats, so C is the named paragraph element. Evidence: PDF p. 15, PDF p. 15, PDF p. 14.

Why each choice is right or wrong

A - Incorrect: Heading, not paragraph.

B - Incorrect: Generic container is not the specified paragraph element.

C - Correct: Exact paragraph element.

D - Incorrect: Anchor/link, not paragraph.

Primary note proof

Study-note section WEB06

> marks paragraphs;
 is a line break and <hr> a thematic break.

Study-note section WEB06

> Heading levels h1-h6 express hierarchy, with h6 conventionally smallest in default styling, not a mandatory CSS size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 15 | lines 27-30

> Paragraph: definition of paragraph
> Format: ...
> E.g., this is a paragraph of text. it has a new line before and after it.
> The browser inserts a new line before and after the text in the paragraph tag.

Web Programming module | PDF page 15 | lines 5-10

> Headings -
> Headings are predefined formats for text presentation and there are six heading formats defined in HTML:
> <h1> up to <h6>
> <h1> the largest font size
> <h6> the smallest font size

Web Programming module | PDF page 14 | lines 5-9

> In HTML, a link is created by wrapping the text or other content inside an <a> tag (anchor tag) and using the href attribute, also known as Hypertext reference, or target that contains the web address.
> Here is an example of a link to the google website
> Google.

Question sources

Computer Science MCQ practice collection · Question 93

Question 204 | AM0094

Which HTML element is used to create an unordered list in a web page?

A.

B.

C.

D. <a>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer A:

The module names ul unordered lists and shows li nested inside it. ol is ordered, while a provides hyperlinks. Evidence: PDF p. 18, PDF p. 19, PDF p. 19, PDF p. 14.

Why each choice is right or wrong

A - Correct: Unordered-list container.

B - Incorrect: List item rather than container.

C - Incorrect: Ordered-list container.

D - Incorrect: Link rather than list container.

Primary note proof

Study-note section WEB08

> Unordered <ul> and ordered <ol> contain <li> items.

Study-note section WEB07

> Use Read for a link; href is the destination, not url/link.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 18 | lines 38-40

> 2.2.8 Ordered and Unordered List in HTML
> Lists
> I. Unordered Lists (ul) - define bulleted lists

Web Programming module | PDF page 19 | lines 2-5

>
> ...

......

Web Programming module | PDF page 19 | lines 14-20

II. Ordered Lists (ol) - element represents an ordered list of items - typically rendered as a numbered list.
Format:

...
......

Web Programming module | PDF page 14 | lines 5-9

In HTML, a link is created by wrapping the text or other content inside an <a> tag (anchor tag) and using the href attribute, also known as Hypertext reference, or target that contains the web address.
Here is an example of a link to the google website
Google.

Question sources

Computer Science MCQ practice collection · Question 94

Question 205 | AM0095

Which HTML element is used to create a link in a web page?

A. <h1>

B.

C. <p>

D. <a>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer D: <a>

An HTML hyperlink wraps its text in an a element with href (p.14). D names that anchor element. Evidence: PDF p. 14, PDF p. 15, PDF p. 15.

Why each choice is right or wrong

A - Incorrect: Heading, not link element.

B - Incorrect: Not the documented anchor element.

C - Incorrect: Paragraph, not anchor.

D - Correct: Exact anchor/link element.

Primary note proof

Study-note section WEB07

> Use Read for a link; href is the destination, not url/link.

Study-note section WEB06

> marks paragraphs;
 is a line break and <hr> a thematic break.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 14 | lines 5-9

> In HTML, a link is created by wrapping the text or other content inside an <a> tag (anchor tag) and using the href attribute, also known as Hypertext reference, or target that contains the web address.
> Here is an example of a link to the google website
> Google.

Web Programming module | PDF page 15 | lines 5-10

> Headings -
> Headings are predefined formats for text presentation and there are six heading formats defined in HTML:
> <h1> up to <h6>
> <h1> the largest font size
> <h6> the smallest font size

Web Programming module | PDF page 15 | lines 27-30

> Paragraph: definition of paragraph
> Format: ...
> E.g., this is a paragraph of text. it has a new line before and after it.
> The browser inserts a new line before and after the text in the paragraph tag.

Question sources

Computer Science MCQ practice collection · Question 95

Question 206 | AM0096

Which CSS property is used to control the spacing between elements in a web page?

A. color

B. font-size

C. margin

D. padding

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer C: margin

The margin is expressly whitespace between a box and other elements, outside border and padding. Padding surrounds content within the box, so C fits spacing between elements. Evidence: PDF p. 36, PDF p. 34.

Why each choice is right or wrong

A - Incorrect: Foreground color, not spacing.

B - Incorrect: Text size, not external spacing.

C - Correct: Exact outer whitespace between elements.

D - Incorrect: Internal whitespace around content, not outer margin.

Primary note proof

Study-note section WEB13

> color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Study-note section WEB13

> Margin lies outside the border; padding lies inside; border surrounds content/padding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 36 | lines 26-37

> The Box Model defines how the different parts of a box - margin, border, padding, and content - work together to create a box that you can see on a page.
> Parts of a box
> - Content box: The area where your content is displayed, which can be sized using properties like width and height.
> - Padding box: The padding sits around the content as white space; its size can be controlled using padding and related properties.
> - Border box: The border box wraps the content and any padding. Its size and style can be controlled using border and related properties.
> - Margin box: The margin is the outermost layer, wrapping the content, padding, and border as whitespace between this box and other elements. Its size can be controlled using margin and related properties.

Web Programming module | PDF page 34 | lines 20-26

> The color property sets the color of the foreground content of the selected elements
> 3.3.2 Font and Text Properties

The font CSS shorthand property sets all the different properties of an element's font. Alternatively, it sets an element's font to a system font. The property is a shorthand for the following properties
- font-family
- font-size

Question sources

Computer Science MCQ practice collection · Question 96

Question 207 | AM0097

Which CSS property is used to change the color of text in a web page?

A. color

B. font-size

C. margin

D. padding

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer A: color

The module explicitly defines color as setting foreground content color. Font size controls dimensions; margin/padding control whitespace. Evidence: PDF p. 34, PDF p. 36.

Why each choice is right or wrong

A - Correct: Exact foreground/text color property.

B - Incorrect: Size rather than color.

C - Incorrect: External whitespace rather than color.

D - Incorrect: Internal whitespace rather than color.

Primary note proof

Study-note section WEB13

color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Study-note section WEB13

Margin lies outside the border; padding lies inside; border surrounds content/padding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 34 | lines 20-26

> The color property sets the color of the foreground content of the selected elements
> 3.3.2 Font and Text Properties
> The font CSS shorthand property sets all the different properties of an element's font. Alternatively, it sets an element's font to a system font. The property is a shorthand for the following properties
> - font-family
> - font-size

Web Programming module | PDF page 36 | lines 31-37

> - Padding box: The padding sits around the content as white space; its size can be controlled using padding and related properties.
> - Border box: The border box wraps the content and any padding. Its size and style can be controlled using border and related properties.
> - Margin box: The margin is the outermost layer, wrapping the content, padding, and border as whitespace between this box and other elements. Its size can be controlled using margin and related properties.

Question sources

Computer Science MCQ practice collection · Question 97

Question 208 | AM0098

Which CSS property is used to change the size of text in a web page?

A. color

B. font-size

C. margin

D. padding

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer B: font-size

Font-size is explicitly listed as the font-size component, required in a multi-property font shorthand. B controls text size; the alternatives control color or whitespace. Evidence: PDF p. 34, PDF p. 36.

Why each choice is right or wrong

A - Incorrect: Foreground color instead of size.

B - Correct: Exact size component.

C - Incorrect: External whitespace instead of text dimensions.

D - Incorrect: Internal whitespace instead of text dimensions.

Primary note proof

Study-note section WEB13

color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Study-note section WEB13

Margin lies outside the border; padding lies inside; border surrounds content/padding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 34 | lines 20-35

The color property sets the color of the foreground content of the selected elements
3.3.2 Font and Text Properties
The font CSS shorthand property sets all the different properties of an element's font. Alternatively, it sets an element's font to a system font. The property is a shorthand for the following properties
- font-family
- font-size
- font-stretch
- font-style
- font-variant
- font-weight
- line-height
If font is specified as a shorthand for several font-related properties, then:
- it must include values for:
<font-size>
<font-family>

Web Programming module | PDF page 36 | lines 31-37

- Padding box: The padding sits around the content as white space; its size can be controlled using padding and related properties.
- Border box: The border box wraps the content and any padding. Its size and style can be controlled using border and related properties.
- Margin box: The margin is the outermost layer, wrapping the content, padding, and border as whitespace between this box and other elements. Its size can be controlled using margin and related properties.

Question sources

Computer Science MCQ practice collection · Question 98

Question 209 | AM0099

In an ordinary horizontal-writing table layout, which property vertically aligns content within a table cell?

A. text-align

B. vertical-align

C. align-content

D. justify-content

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer B: vertical-align

vertical-align applies in table-cell and inline contexts. With an ordinary horizontal-writing table cell specified, it controls the vertical placement of the cell's content. text-align handles inline-content text alignment, while flex or grid alignment properties have their own axis and line-distribution contexts. The table-cell qualifier is essential because vertical-align does not vertically center arbitrary block containers simply because they contain content.

Why each choice is right or wrong

A - Incorrect: text-align handles inline-content alignment rather than vertical cell placement.

B - Correct: vertical-align applies to the specified table-cell context.

C - Incorrect: align-content distributes applicable lines or tracks, not ordinary table-cell content.

D - Incorrect: justify-content follows a flex/grid main-axis context rather than this table-cell mechanism.

Primary note proof

Study-note section WEB14

> vertical-align applies to inline/table-cell contexts and does not center every block.

Study-note section WEB14

> Flexbox arranges direct children on a main axis; justify-content follows that axis, align-items the cross axis.

Study-note section WEB13

> color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 35 | lines 15-21

> The text-align property is used to control how text is aligned within its containing content box. Values
> - left: Left-justifies the text.
> - right: Right-justifies the text.
> - center: Centers the text.
> - justify: Makes the text spread out, varying the gaps in between the words so that all lines of text are the same width.

Web Programming module | PDF page 37 | lines 15-18

> Display
> The display CSS property sets whether an element is treated as a block or inline element and the layout used for its children, such as flow layout, grid or flex. Formally, the display property sets an element's inner and outer display types. The

Question sources

Computer Science MCQ practice collection · Question 99

Question 210 | AM0100

Which CSS property is used to create a border around an element in a web page?

A. color

B. font-size

C. border

D. padding

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer C: border

The border box wraps content and padding and is controlled with border properties (p.36). The table section also explicitly says border sets borders. Evidence: PDF p. 36, PDF p. 38.

Why each choice is right or wrong

A - Incorrect: Foreground color rather than border.

B - Incorrect: Text size rather than border.

C - Correct: Exact enclosing-border property.

D - Incorrect: Whitespace inside the border, not the border itself.

Primary note proof

Study-note section WEB13

> color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Study-note section WEB13

> Margin lies outside the border; padding lies inside; border surrounds content/padding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 36 | lines 29-37

> - Content box: The area where your content is displayed, which can be sized using properties like width and height.
> - Padding box: The padding sits around the content as white space; its size can be controlled using padding and related properties.
> - Border box: The border box wraps the content and any padding. Its size and style can be controlled using border and related properties.
> - Margin box: The margin is the outermost layer, wrapping the content, padding, and border as whitespace between this box and other elements. Its size can be controlled using margin and related properties.

Web Programming module | PDF page 38 | lines 19-24

> - border-collapse => By default, when you set borders on table elements, they will all have spacing between them. To prevent that from happening we can use the border-collapse CSS property and set its value to collapse.
> - We can set borders to using the border property, we can also apply the following properties depending on out preference => padding, font, text-align, text-shadow, color, background-color and others

Question sources

Computer Science MCQ practice collection · Question 100

Question 211 | AM0101

Which CSS property is used to control the space inside the border of an element in a web page?

A. color

B. font-size

C. margin

D. padding

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer D: padding

Padding sits around the content as whitespace; the border wraps that padding. Therefore the internal space inside an element border is padding. Evidence: PDF p. 36.

Why each choice is right or wrong

A - Incorrect: Foreground color, not space.

B - Incorrect: Text dimensions, not space.

C - Incorrect: Outermost whitespace beyond the border.

D - Correct: Exact whitespace surrounding content inside border.

Primary note proof

Study-note section WEB13

> color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Study-note section WEB13

> Margin lies outside the border; padding lies inside; border surrounds content/padding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 36 | lines 29-37

> - Content box: The area where your content is displayed, which can be sized using properties like width and height.
> - Padding box: The padding sits around the content as white space; its size can be controlled using padding and related properties.
> - Border box: The border box wraps the content and any padding. Its size and style can be controlled using border and related properties.
> - Margin box: The margin is the outermost layer, wrapping the content, padding, and border as whitespace between this box and other elements. Its size can be controlled using margin and related properties.

Question sources

Computer Science MCQ practice collection · Question 101

Question 212 | AM0102

Which web technology is commonly used for managing the content of information-rich websites?

A. HTML

B. CSS

C. JavaScript

D. Content Management System (CMS)

Hint

Distinguish the purpose of each development or performance technique from its possible side effects.

Answer D: Content Management System

A CMS gives editors tools to create, publish, and manage content without editing each source file. That is the technology aimed at information-rich website content management. HTML structures documents, CSS styles them, and JavaScript adds behavior, but those technologies alone do not supply the complete editorial management interface described.

Why each choice is right or wrong

A - Incorrect: HTML structures document content but does not by itself provide an editorial management workflow.

B - Incorrect: CSS controls presentation rather than managing editorial content creation, publication, and roles.

C - Incorrect: JavaScript can implement behavior but is not itself the complete content-management system requested.

D - Correct: A CMS provides editor-facing creation, publishing, role, and media tools for managing website content.

Primary note proof

Study-note section WEB39

> A CMS gives editors content creation, publishing, roles and media management without editing every source file.

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 10 | lines 9-16

HTML (HyperText Markup Language) is the most basic building block of the Web. HTML is not a programming language rather it is a markup language that tells web browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Web Programming module | PDF page 9 | lines 13-21

Web hosting is the place where all the files of your website live. Web hosting is a service that allows organizations and individuals to post a website or a web page onto the Internet. A web host, or web hosting service provider, is a business that provides the technologies and services needed for the website or webpage to be viewed in the Internet. Websites are hosted, or stored, on special computers called servers. When Internet users want to view your website, all they need to do is type your website address or domain (the domain name for the site has to be registered) into their browser. Their computer will then connect to your server and your webpages will be delivered to them through the browser.

Question sources

Computer Science MCQ practice collection · Question 102

Question 213 | AM0103

What is the database’s direct role in a web application?

A. To control the visual appearance of web pages

B. To store persistent application data and support queries over that data

C. To specify the structure and content of web pages

D. To add interactivity and dynamic behavior to web pages

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer B: To store persistent application data and support queries over that data

A database stores persistent structured application facts and supports operations that retrieve or change them. It is not the whole application server, which also performs validation, authorization and response generation. HTML describes document structure, CSS controls visual presentation and browser scripting supplies client interaction. The option states the database’s actual storage/query role rather than attributing all server-side processing to it.

Why each choice is right or wrong

A - Incorrect: Visual presentation is the role of CSS.

B - Correct: Persistent data storage and queries are the database's direct role.

C - Incorrect: Document structure is the role of HTML.

D - Incorrect: Client-side interactivity is a scripting role rather than the database's direct purpose.

Primary note proof

Study-note section WEB31

> MySQL is a relational DBMS; a database persists structured application data and supports concurrent clients.

Study-note section WEB39

> A database stores persistent facts; it is not itself the whole server application.

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 78 | lines 24-31

> MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. It is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.
> - MySQL is a database management system.
> A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network.

Web Programming module | PDF page 60 | lines 11-16

> Client-side scripting is the glamorous, eye-catching part of Web development. In contrast, server-side scripting is invisible to the user. Server-side Web scripting is mostly about connecting Web sites to back-end servers, such as databases. This enables the following types of two-way communication:
> ➢ Server to client: Web pages can be assembled from back end-server output.
> ➢ Client to server: Customer-entered information can be acted upon.

Web Programming module | PDF page 10 | lines 14-16

> HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Question sources

Computer Science MCQ practice collection · Question 103

Question 214 | AM0104

Which SQL statement is used to retrieve data from a database?

A. INSERT INTO

B. UPDATE

C. DELETE FROM

D. SELECT

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer D: SELECT

The SELECT query example selects all student records and then fetches them. INSERT, UPDATE and DELETE are listed as other commands, not the demonstrated retrieval operation. Evidence: PDF p. 80, PDF p. 80.

Why each choice is right or wrong

A - Incorrect: Not the retrieval command used in the example.

B - Incorrect: Not the retrieval command used in the example.

C - Incorrect: Not the retrieval command used in the example.

D - Correct: SELECT is used before fetching records.

Primary note proof

Study-note section WEB32

> SELECT retrieves rows; INSERT adds; UPDATE ...

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 80 | lines 2-15

> 5.13.3 Sending Query to MySQL Database using PHP
> A database query from PHP is basically a MySQL command wrapped up in a tiny
> PHP function called mysql_query(). This is where you use the basic SQL workhorses
> of SELECT,
> INSERT, UPDATE, and DELETE, CREATE or DROP a table (but not those to create
> or drop an entire database) can be used with this PHP function.
> Syntax:
> [$resultvar=] mysql_query(“query”,[link identifier]);
> $resultvar - is an optional result variable. This is used when the query is a select
> statement.
> Query - is any valid insert, delete, update, select, create table or drop table
> statement enclosed by quotes. It is a good habit to hold the query in a separate variable

to add flexibility to your code.
Link identifier-is an optional link identifier

Web Programming module | PDF page 80 | lines 24-29

Example: assume the student table has three attrubutes - id,name and department.
$query=”select * from student”;
$result=mysql_query($query);
while($row=mysql_fetch_row($result)){
echo “$row[0].$row[1].$row[2]”;//counting
starts from 0 not 1

Question sources

Computer Science MCQ practice collection · Question 104

Question 215 | AM0105

A server starts by executing the following CommonJS source directly and registers a callback for incoming HTTP requests:
const http = require("node:http");
http.createServer((req, res) => { res.end("Ready"); }).listen(8000);
Which listed programming language is executing this request handler?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer C: JavaScript

The CommonJS server runs in Node.js, which is a JavaScript runtime rather than a separate programming language. The require call loads the HTTP module, and the callback passed to createServer handles incoming requests and ends each response. PHP can also generate server responses, but this particular source uses the Node JavaScript API and syntax. HTML and CSS do not execute the handler as a server program.

Why each choice is right or wrong

A - Incorrect: HTML supplies document structure rather than executing the module load and request callback.

B - Incorrect: CSS supplies presentation rules rather than this executable request handler.

C - Correct: Node.js executes this CommonJS request handler as JavaScript.

D - Incorrect: PHP supports server programming but does not directly execute this CommonJS source.

Primary note proof

Study-note section WEB01

> Node.js is a JavaScript runtime, not a separate language.

Study-note section WEB36

> A minimal CommonJS Node server can use const http=require("node:http"); http.createServer((req,res)=>{res.writeHead(200,{"Content-Type":"text/plain"});res.end("Hello");}).listen(8000);.

Study-note section WEB36

> The request listener runs for incoming requests; end completes the response.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 60 | lines 4-10

> PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Web Programming module | PDF page 92 | lines 6-9

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.

Web Programming module | PDF page 10 | lines 9-16

> HTML (HyperText Markup Language) is the most basic building block of the Web. HTML is not a programming language rather it is a markup language that tells web browsers how to structure the web page. HTML consists of a series of elements, which you use to enclose, wrap or markup different parts of content to make it appear or act in a certain way.
> HTML defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/ behavior (JavaScript).

Question sources

Computer Science MCQ practice collection · Question 105

Computer Science MCQ practice collection · Question 123

Question 216 | AM0106

Which HTML form element is used to create a text input field?

A. <input type="text">

B. <input type="checkbox">

C. <input type="radio">

D. <input type="submit">

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer A: <input type="text">

The input type list identifies text as the usual text input and distinguishes checkbox/radio. The example on p.56 uses input type=text with a Name field. Evidence: PDF p. 22, PDF p. 56, PDF p. 22.

Why each choice is right or wrong

A - Correct: Exact text-input type.

B - Incorrect: Checkbox type instead.

C - Incorrect: Radio type instead.

D - Incorrect: Button submission type instead of text entry.

Primary note proof

Study-note section WEB-S02

> In an input element, type="text" creates text entry, type="checkbox" an independently selectable checkbox, type="radio" a radio choice, and type="submit" a submit button.

Study-note section WEB09

> Same-name radio buttons form a single-choice group; checkboxes permit independent selections.

Study-note section WEB09

> submit submits; button has no submission action by itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 22 | lines 2-9

> 1. Text inputs create using the <input> tag, an input control can take many forms
> using the type attribute. Possible values for type attribute
> - Text - most common used for text inputs
> - Password - for password fields

- Checkbox
- Radio
Other attributes for <input> tag
- Name, value, size, maxlength

Web Programming module | PDF page 56 | lines 2-2

<input type = "text" name = "Name" />

Web Programming module | PDF page 22 | lines 21-24

5. Button controls → we can control various types of buttons in HTML. We can create a clickable button by using the <input> tag and setting the type attribute to one of the following values
- Submit, reset, button, image

Question sources

Computer Science MCQ practice collection · Question 106

Question 217 | AM0107

Which HTML form element is used to create a checkbox?

A. <input type="text">

B. <input type="checkbox">

C. <input type="radio">

D. <input type="submit">

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer B: <input type="checkbox">

The module input type list includes Checkbox separately from Text and Radio. Therefore input type=checkbox is the checkbox control, not a text field, radio choice or submit button. Evidence: PDF p. 22, PDF p. 22.

Why each choice is right or wrong

A - Incorrect: Text entry control.

B - Correct: Exact checkbox type.

C - Incorrect: Radio control instead.

D - Incorrect: Submit button instead.

Primary note proof

Study-note section WEB-S02

In an input element, type="text" creates text entry, type="checkbox" an independently selectable checkbox, type="radio" a radio choice, and type="submit" a submit button.

Study-note section WEB09

Same-name radio buttons form a single-choice group; checkboxes permit independent selections.

Study-note section WEB09

submit submits; button has no submission action by itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 22 | lines 2-9

1. Text inputs create using the <input> tag, an input control can take many forms using the type attribute. Possible values for type attribute
- Text - most common used for text inputs
- Password - for password fields
- Checkbox
- Radio
Other attributes for <input> tag
- Name, value, size, maxlength

Web Programming module | PDF page 22 | lines 21-24

5. Button controls → we can control various types of buttons in HTML. We can create a clickable button by using the <input> tag and setting the type attribute to one of the following values
- Submit, reset, button, image

Question sources

Computer Science MCQ practice collection · Question 107

Question 218 | AM0108

Which HTML form element is used to create a radio button?

A. <input type="text">

B. <input type="checkbox">

C. <input type="radio">

D. <input type="submit">

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer C: <input type="radio">

The module distinguishes the Radio input type from Text and Checkbox. C therefore specifies a radio button. Evidence: PDF p. 22, PDF p. 22.

Why each choice is right or wrong

A - Incorrect: Text entry instead.

B - Incorrect: Checkbox instead.

C - Correct: Exact radio input type.

D - Incorrect: Submit button instead.

Primary note proof

Study-note section WEB-S02

> In an input element, type="text" creates text entry, type="checkbox" an independently selectable checkbox, type="radio" a radio choice, and type="submit" a submit button.

Study-note section WEB09

> Same-name radio buttons form a single-choice group; checkboxes permit independent selections.

Study-note section WEB09

> submit submits; button has no submission action by itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 22 | lines 2-9

> 1. Text inputs create using the <input> tag, an input control can take many forms using the type attribute. Possible values for type attribute
> - Text - most common used for text inputs
> - Password - for password fields
> - Checkbox
> - Radio
> Other attributes for <input> tag
> - Name, value, size, maxlength

Web Programming module | PDF page 22 | lines 21-24

> 5. Button controls → we can control various types of buttons in HTML. We can create a clickable button by using the <input> tag and setting the type attribute to one of the following values
> - Submit, reset, button, image

Question sources

Computer Science MCQ practice collection · Question 108

Question 219 | AM0109

Which HTML form element is used to create a button that submits a form?

A. <input type="text">

B. <input type="checkbox">

C. <input type="radio">

D. <input type="submit">

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer D: <input type="submit">

The button-controls list includes Submit as an input type. Combined with the form submission description, D is the form-submitting button; the others collect text or selections. Evidence: PDF p. 22, PDF p. 21, PDF p. 22.

Why each choice is right or wrong

A - Incorrect: Text input, not submission button.

B - Incorrect: Checkbox selection, not submission button.

C - Incorrect: Radio selection, not submission button.

D - Correct: Submit button as listed in the controls section.

Primary note proof

Study-note section WEB-S02

> In an input element, type="text" creates text entry, type="checkbox" an independently selectable checkbox, type="radio" a radio choice, and type="submit" a submit button.

Study-note section WEB09

> Same-name radio buttons form a single-choice group; checkboxes permit independent selections.

Study-note section WEB09

> submit submits; button has no submission action by itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 22 | lines 21-24

> 5. Button controls → we can control various types of buttons in HTML. We can create a clickable button by using the <input> tag and setting the type attribute to one of the following values
> - Submit, reset, button, image

Web Programming module | PDF page 21 | lines 13-20

> 2.2.10 HTML Form and Form Controls
> The <form> HTML element represents a document section containing interactive controls for submitting information. An HTML form is a section of a document containing normal content, markup, special elements called controls (checkboxes, radio buttons, menus, etc.) and labels on those controls. HTML Forms are required, when we want to collect data from a user. Using forms, we can build a website with user registration. We take the information collected from the user and then post it to the backend application such as php.

Web Programming module | PDF page 22 | lines 2-7

> 1. Text inputs create using the <input> tag, an input control can take many forms using the type attribute. Possible values for type attribute
> - Text - most common used for text inputs
> - Password - for password fields
> - Checkbox
> - Radio

Question sources

Computer Science MCQ practice collection · Question 109

Question 220 | AM0110

Which client-side programming language can run a custom submit handler that compares two form fields and cancels submission when they disagree?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Separate event dispatch, handler registration and cancellation of the browser default action.

Answer C: JavaScript

JavaScript can run browser event handlers, inspect field values and call preventDefault when a custom validity check fails. This supports a comparison between fields before submission. HTML supplies native constraints such as required or type=email, but those constraints are not the custom scripted comparison requested. CSS controls presentation, and PHP processes server requests after submission rather than running the browser handler.

Why each choice is right or wrong

A - Incorrect: HTML supplies native field constraints, but is not the programming language executing the custom comparison handler.

B - Incorrect: CSS controls presentation rather than executing submit-handler logic.

C - Correct: JavaScript executes the browser submit handler and can cancel the default submission on a failed comparison.

D - Incorrect: PHP runs the server-side processing, not this browser event handler.

Primary note proof

Study-note section WEB21

> For submission, form.addEventListener("submit",e=>{if(!valid()){e.preventDefault();}}) stops the default only on failure.

Study-note section WEB09

> type=email/required give browser validation feedback; arbitrary clients can bypass it.

Study-note section WEB01

> HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

> PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 55 | lines 4-18

> Now inside the submit event handler function we call the event.preventDefault()
> method to prevent the default behavior and avoid a form submit to reload the page,
> this gives us control. After we gain control over the submit event we can all kind of
> stuff, like form validation
> Here is a simple but different way of validating a form using javascript
> Form Validation
> <html>
> <head>
> <title>Form Validation</title>
> <script type = "text/javascript">
> if(document.myForm.Name.value == "") {
> alert("Please provide your name!");
> document.myForm.Name.focus() ;
> return false;
> }

Web Programming module | PDF page 73 | lines 2-7

> Form validation is done to ensure that the user has provided the relevant information.
> Basic validation can be done using HTML elements. For example, in the above
> script, the email address text box is having a type value as “email”, which prevents

the user from entering the incorrect value for an email. Every form field in the above script is followed by a required attribute, which will intimate the user not to leave any field empty before submitting the form. PHP methods and arrays used in form

Web Programming module | PDF page 72 | lines 25-33

5.10 Form Processing using PHP
Handling a form is a multipart process. First a form is created, into which a user can enter the required details. This data is then sent to the web server, where it is interpreted,
often with some error checking. If the PHP code identifies one or more fields that require
reentering, the form may be redisplayed with an error message. When the code is satisfied with the accuracy of the input, it takes some action that usually involves the database.

Question sources

Computer Science MCQ practice collection · Question 110

Question 221 | AM0111

Which description defines client-server roles without requiring a particular physical-machine arrangement?

A. Client and server must be the same physical machine

B. A client requests resources or services and a server receives requests and returns responses; they may share a machine

C. Any two different browser instances automatically form client and server roles

D. Any two pages on a website automatically form client and server roles

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: A client requests resources or services and a server receives requests and returns responses; they may share a machine

Client and server describe interacting roles: the client requests a resource or service and the server handles the request and returns a response. These roles can run on separate machines or on the same machine. Physical location therefore does not define the relationship. Two browser instances or two web pages also do not automatically establish a request-serving role pair. The corrected definition focuses on behavior while retaining the deployment distinction as a distractor.

Why each choice is right or wrong

A - Incorrect: The roles can share a machine, but sameness is not required.

B - Correct: This states the request/response roles and permits either physical arrangement.

C - Incorrect: Two browser instances do not automatically make one a server.

D - Incorrect: Different pages are resources, not automatically request and serving roles.

Primary note proof

Study-note section WEB01

> A browser/user agent requests resources and renders HTML/CSS, then runs permitted JavaScript.

Study-note section WEB01

> A web/HTTP server receives requests and returns representations, sometimes invoking application code.

Study-note section WEB01

> Client and server are roles and can run on the same machine.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 8 | lines 29-34

> • The World Wide Web (WWW) is most often called the Web.
> • The Web is a network of computers all over the world.
> • All the computers in the Web can communicate with each other.
> • All the computers use a communication standard called HTTP.
> How does the WWW work?
> • Web information is stored in documents called Web pages.

Web Programming module | PDF page 9 | lines 2-4

> • Web pages are files stored on computers called Web servers.
> • Computers reading the Web pages are called Web clients.
> • Web clients view the pages with a program called a Web browser.

Question sources

Computer Science MCQ practice collection · Question 111

Question 222 | AM0112

A browser sends a GET request and the web server replies with a status code and resource representation. Which application-layer protocol gives this exchange its semantics?

A. HTTP

B. FTP

C. TCP/IP

D. SMTP

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: HTTP

GET and the returned status code belong to HTTP's web request/response semantics. TCP/IP may carry that exchange, but it does not define the application method or response-status meaning. FTP and SMTP serve different application purposes. The scenario therefore identifies HTTP without depending on an unstated assumption about which networking layer is being tested.

Why each choice is right or wrong

A - Correct: HTTP defines GET requests, response status codes and resource representations.

B - Incorrect: FTP does not define the stated web GET/status exchange.

C - Incorrect: TCP/IP can carry the exchange but does not define its application-level GET or status semantics.

D - Incorrect: SMTP is a mail-transfer application protocol, not the stated web request/response semantics.

Primary note proof

Study-note section WEB01

> The Internet connects networks; the Web is a resource system commonly using HTTP.

Study-note section WEB-S14

> HTTP defines web application requests, response status codes and representations.

Study-note section WEB-S14

> An HTTP exchange can use underlying TCP connections and IP addresses; naming that network stack does not identify the application-level method or status semantics.

Study-note section WEB-S14

> FTP is a file-transfer application protocol.

Study-note section WEB-S14

> SMTP transports electronic mail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 8 | lines 29-34

> • The World Wide Web (WWW) is most often called the Web.
> • The Web is a network of computers all over the world.
> • All the computers in the Web can communicate with each other.
> • All the computers use a communication standard called HTTP.

How does the WWW work?
• Web information is stored in documents called Web pages.

Question sources

Computer Science MCQ practice collection · Question 112

Question 223 | AM0113

Which method in HTTP is used to retrieve a resource from a server?

A. GET

B. POST

C. PUT

D. DELETE

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer A: GET

WEB04 defines GET as retrieving a representation with safe semantics. That is the retrieval operation requested. POST asks a target to process supplied content, PUT creates or replaces the identified representation, and DELETE requests removal. These are intended method semantics; a badly designed server does not change which method the standard retrieval role belongs to.

Why each choice is right or wrong

A - Correct: GET is the HTTP method whose intended semantics retrieve a resource representation.

B - Incorrect: POST requests processing of supplied content rather than naming the standard safe retrieval operation.

C - Incorrect: PUT creates or replaces an identified representation instead of simply retrieving it.

D - Incorrect: DELETE requests removal of a resource association rather than reading its representation.

Primary note proof

Study-note section WEB04

GET retrieves a representation with safe semantics; do not implement an intentional state-changing action as a GET link.

Study-note section WEB04

POST asks the target to process supplied content.

Study-note section WEB04

PUT creates/replaces the identified resource representation; DELETE requests removal of its association.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 25-31

In order to make an HTTP request to the server with JavaScript, you need an instance of an XMLHttpRequest Object. We make a request to the sever, by calling the open() and send() methods of the HTTP request object, like this:
httpRequest.open('GET', 'http://www.example.org/some.file', true);
httpRequest.send();
open() takes three parameters, the first is the HTTP request method and the second is the URL. The optional third parameter sets whether the request is asynchronous.

Web Programming module | PDF page 73 | lines 11-16

• $_GET[]: It is used the retrieve the information from the form control through the parameters sent in the URL. It takes the attribute given in the url as the parameter.
• $_POST[]: It is used the retrieve the information from the form control through the HTTP POST method. IT takes name attribute of corresponding form control as the parameter.

Question sources

Computer Science MCQ practice collection · Question 113

Question 224 | AM0114

Which method in HTTP is used to send data to a server for processing?

A. GET

B. POST

C. PUT

D. DELETE

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer B: POST requests processing of supplied content.

WEB04 explicitly says POST asks the target to process supplied content, matching the intended HTTP method role. GET retrieves safely, PUT creates or replaces an identified representation, and DELETE removes an association. Those methods can carry protocol details, but the generic processing semantics in the stem are POST's specific definition.

Why each choice is right or wrong

A - Incorrect: GET's intended semantics retrieve a representation rather than request arbitrary content processing.

B - Correct: POST asks the target resource to process the supplied content.

C - Incorrect: PUT specifies creating or replacing the identified representation rather than the generic processing role.

D - Incorrect: DELETE requests removal of a resource association rather than the stated content-processing semantics.

Primary note proof

Study-note section WEB04

> GET retrieves a representation with safe semantics; do not implement an intentional state-changing action as a GET link.

Study-note section WEB04

> POST asks the target to process supplied content.

Study-note section WEB04

> PUT creates/replaces the identified resource representation; DELETE requests removal of its association.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 73 | lines 11-16

> • $_GET[]: It is used the retrieve the information from the form control through the parameters sent in the URL. It takes the attribute given in the url as the parameter.
> • $_POST[]: It is used the retrieve the information from the form control through the HTTP POST method. IT takes name attribute of corresponding form control as the parameter.

Web Programming module | PDF page 21 | lines 23-31

> Form attributes:
> - action → Backend script ready to process the data collected
> - method → method used to upload Data. The most frequently used
> - target → Specify the target window or frame where the result of the script will be displayed. It takes values like _blank, _self, _parent etc.
> - enctype → tells how the browser encodes data. Possible values
> 1. application/x-www-form-urlencoded → most common and used from simple scenarios
> 2. mutlipart/form-data → used to upload binary data

Question sources

Computer Science MCQ practice collection · Question 114

Question 225 | AM0115

Which HTTP status code indicates a successful request?

A. 200

B. 300

C. 400

D. 500

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer A: 200

WEB04 explicitly maps 200 to success, 400 to a malformed request, and 500 to unexpected server failure. A 3xx code belongs to redirection rather than ordinary successful representation delivery. Therefore 200 is the listed status indicating a successful request. Other 2xx successes exist, but they are not among these options.

Why each choice is right or wrong

A - Correct: The note explicitly identifies 200 as a successful request status.

B - Incorrect: A 3xx code represents redirection behavior rather than the listed ordinary success status.

C - Incorrect: 400 indicates a malformed request, not successful processing.

D - Incorrect: 500 indicates an unexpected server failure, not success.

Primary note proof

Study-note section WEB04

> 200 indicates success, 201 creation, 204 success without content, 301 permanent redirection, 400 malformed request, 401 missing/invalid authentication, 403 refusal, 404 unavailable/not found, and 500 an unexpected server failure.

Study-note section WEB-S10

> The first digit of an HTTP status code identifies its class: 2xx means successful handling, 3xx means redirection or further action, 4xx indicates a client-error condition, and 5xx indicates a server-error condition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 92 | lines 24-27

```
const http = require('http');
const port = 8000;
const server = http.createServer((req, res) => {
res.statusCode = 200;
```

Web Programming module | PDF page 93 | lines 2-4

```
res.setHeader('Content-Type', 'text/plain');
res.end('Hello World');
});
```

Question sources

Computer Science MCQ practice collection · Question 115

Question 226 | AM0116

Which HTTP status code indicates that the requested resource is not found on the server?

A. 200

B. 300

C. 400

D. 404

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer D: 404

WEB04 explicitly identifies 404 as unavailable or not found. That directly matches the requested missing-resource case. The alternatives represent success, redirection, or a malformed request rather than the not-found status. A not-found response describes what is reported to the client; it does not require explaining every possible server-side reason for the absence.

Why each choice is right or wrong

A - Incorrect: 200 reports success rather than the missing-resource condition in the stem.

B - Incorrect: 300 is a redirection-class code, not the note's not-found status.

C - Incorrect: 400 concerns a malformed request rather than the unavailable resource identified here.

D - Correct: 404 is explicitly mapped to unavailable or not found in WEB04.

Primary note proof

Study-note section WEB04

> 200 indicates success, 201 creation, 204 success without content, 301 permanent redirection, 400 malformed request, 401 missing/invalid authentication, 403 refusal, 404 unavailable/not found, and 500 an unexpected server failure.

Study-note section WEB-S10

> The first digit of an HTTP status code identifies its class: 2xx means successful handling, 3xx means redirection or further action, 4xx indicates a client-error condition, and 5xx indicates a server-error condition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 92 | lines 24-27

```
const http = require('http');
const port = 8000;
const server = http.createServer((req, res) => {
res.statusCode = 200;
```

Question sources

Computer Science MCQ practice collection · Question 116

Question 227 | AM0117

Which situations demonstrate concurrency when their requests make overlapping progress? Here simultaneously means overlapping progress, not necessarily execution at the same instant. Select all that apply.

A. The ability of the server to handle multiple clients simultaneously

B. The ability of the client to handle multiple servers simultaneously

C. The ability of the server to handle multiple requests from a single client

D. The ability of the client to handle multiple requests to a single server

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer A, B, C, D: The ability of the server to handle multiple clients simultaneously; The ability of the client to handle multiple servers simultaneously; The ability of the server to handle multiple requests from a single client; The ability of the client to handle multiple requests to a single server

Concurrency means overlapping progress, whereas parallelism requires execution at the same instant. A server can overlap work for several clients or several requests from one client. A client can likewise overlap interactions with several servers or multiple requests to one server. The question explicitly supplies overlap for each situation, making every listed arrangement an example of concurrency without asserting that all work physically runs in parallel.

Why each choice is right or wrong

A - Correct: Overlapping server work for several clients is concurrent.

B - Correct: Overlapping client interactions with several servers are concurrent.

C - Correct: Several overlapping requests from one client can create server concurrency.

D - Correct: A client can overlap multiple requests to the same server.

Primary note proof

Study-note section WEB36

> Concurrency means overlapping progress; parallelism means execution at the same instant.

Study-note section WEB36

> Multiple clients, or one client with overlapping requests, can both create concurrency.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 92 | lines 6-14

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.
> Benefits of using Node
> 1. Performance
> 2. Uses plain old JavaScript
> 3. Thousands of reusable packages
> 4. Portable

Question sources

Computer Science MCQ practice collection · Question 117

Question 228 | AM0118

What is a race condition in the context of concurrency?

A. A condition where two or more clients compete to establish a connection with the server

B. A condition where two or more servers compete to respond to a client request

C. A condition where multiple threads or processes access shared resources in an unpredictable manner

D. A condition where multiple clients send duplicate requests to the server

Hint

Identify the shared-state invariant and the operations that must be coordinated.

Answer C: Shared-resource correctness depends on execution interleaving.

WEB37 describes two requests reading stock one, both accepting a sale, and both writing zero. The final correctness depends on their interleaving, which is the race-condition mechanism captured most closely by C's unpredictable shared-resource access. Competing connections, multiple responders, or duplicate requests are not themselves sufficient; they become races only when shared-state correctness depends on timing.

Why each choice is right or wrong

A - Incorrect: Competition to establish a connection is not by itself a shared-state correctness failure caused by interleaving.

B - Incorrect: Several servers responding does not alone establish a race on shared mutable state.

C - Correct: This is the closest option: shared-state access can make correctness depend on the execution interleaving.

D - Incorrect: Duplicate requests can cause other problems, but duplication alone is not the defining race-condition mechanism.

Primary note proof

Study-note section WEB37

> Suppose two requests read a stock count1, both decide it is available and both write0: two sales consumed one item.

Study-note section WEB37

> Correctness depends on interleaving, even if each callback has only a short synchronous body.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 92 | lines 6-14

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.
> Benefits of using Node
> 1. Performance
> 2. Uses plain old JavaScript
> 3. Thousands of reusable packages
> 4. Portable

Question sources

Computer Science MCQ practice collection · Question 118

Question 229 | AM0119

Which technique directly protects one shared-state critical section with an exclusive lock that does not rely on busy waiting? Choose the exclusive-lock primitive, rather than a bounded-access counter or a messaging design.

A. Mutex locks

B. Spinlocks

C. Semaphores

D. Message queues

Hint

Identify the shared-state invariant and the operations that must be coordinated.

Answer A: Mutex locks

A mutex directly protects a critical section by allowing only one holder at a time. The question distinguishes that exclusive-lock abstraction from spinlock busy waiting, semaphore bounded-access counting and message-queue designs. A semaphore can also enforce exclusion when configured suitably, but it is not the specific exclusive-lock primitive requested. Choosing a primitive does not by itself guarantee application correctness: every conflicting update must use the protected critical section.

Why each choice is right or wrong

A - Correct: A mutex is the requested exclusive-lock primitive.

B - Incorrect: A spinlock relies on busy waiting, which the stem excludes.

C - Incorrect: A semaphore is the bounded-access counter abstraction excluded by the requested category.

D - Incorrect: A message queue is a messaging design rather than the requested lock primitive.

Primary note proof

Study-note section WEB37

> A mutex protects one critical section; semaphores can coordinate bounded access; spinlocks busy-wait and are unsuitable for blocking a browser/event-loop thread.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 92 | lines 6-14

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.
> Benefits of using Node
> 1. Performance

2. Uses plain old JavaScript
3. Thousands of reusable packages
4. Portable

Question sources

Computer Science MCQ practice collection · Question 119

Question 230 | AM0120

In an asynchronous browser request, what is the response callback's role?

A. A function invoked by the browser runtime or API when the response becomes available

B. A function that is called by the client to request a specific operation from the server

C. A function that is called by the server to handle concurrent client requests

D. A function that is called by the client to synchronize with the server

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer A: A function invoked by the browser runtime or API when the response becomes available

The callback is registered for later invocation by the browser runtime or API when the response becomes available. The remote server sends response data; it does not directly execute the browser function. A function that starts a request or a server request-handler function has a different role in the described interaction. The answer states both the owning execution environment and the event that causes invocation.

Why each choice is right or wrong

A - Correct: This states who invokes the browser response callback and when.

B - Incorrect: Starting a request is not the response callback's role.

C - Incorrect: A server request handler is a different callback in a different environment.

D - Incorrect: A callback need not perform client/server synchronization.

Primary note proof

Study-note section WEB34

A callback is a function invoked later by the runtime/API.

Study-note section WEB34

The browser invokes its response handler when the response is available; the remote server does not directly execute browser code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 90 | lines 2-10

> 6.4 Handling Response from Server
> Before making a request to the server we have to write a callback function which will handle the response. But Because Ajax calls are asynchronous, we can't be sure when the response will come, so we must write code to wait for the response and handle it when it arrives. That is what the callback function is used for
> At this stage, we need to tell the XMLHttpRequest object which JavaScript function will handle the response, by setting the onreadystatechange property of the object and naming it after the function to call when the request changes state, like this:
> httpRequest.onreadystatechange = nameOfTheFunction;

Question sources

Computer Science MCQ practice collection · Question 120

Question 231 | AM0121

Which application-level protocol provides a persistent bidirectional message channel after its handshake?

A. HTTP

B. FTP

C. TCP/IP

D. WebSocket

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer D: WebSocket

WebSocket establishes a persistent bidirectional channel on which either endpoint can exchange messages after the handshake. HTTP requests can also be issued asynchronously, but asynchronous programming alone does not supply this specific channel model. FTP has a different transfer role, and TCP/IP names lower-level networking foundations rather than the requested application protocol. The revised interaction requirement therefore distinguishes WebSocket without claiming it is the only way to avoid blocking a client.

Why each choice is right or wrong

A - Incorrect: Asynchronous HTTP requests do not by themselves define this persistent bidirectional message channel.

B - Incorrect: FTP is not the specified WebSocket-style application messaging protocol.

C - Incorrect: TCP/IP refers to lower-level networking rather than this application-level channel protocol.

D - Correct: WebSocket supplies the described persistent bidirectional channel.

Primary note proof

Study-note section WEB35

WebSocket establishes a persistent bidirectional message channel after its handshake.

Study-note section WEB35

Either HTTP-based AJAX or WebSockets can support nonblocking user interaction and carry JSON, so generic questions may be nonexclusive.

Study-note section WEB-S14

HTTP defines web application requests, response status codes and representations.

Study-note section WEB-S14

An HTTP exchange can use underlying TCP connections and IP addresses; naming that network stack does not identify the application-level method or status semantics.

Study-note section WEB-S14

FTP is a file-transfer application protocol.

Study-note section WEB-S14

SMTP transports electronic mail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 20-31

6.3 Sending Request to PHP server
XMLHttpRequest (XHR) is an API that can be used by JavaScript and other web browser scripting languages to transfer and manipulate XML data to and from a webserver using HTTP, establishing an independent connection channel between a webpage's Client-Side and Server-Side.
In order to make an HTTP request to the server with JavaScript, you need an instance of an XMLHttpRequest Object. We make a request to the sever, by calling the open() and send() methods of the HTTP request object, like this:
httpRequest.open('GET', 'http://www.example.org/some.file', true);
httpRequest.send();
open() takes three parameters, the first is the HTTP request method and the second is the URL. The optional third parameter sets whether the request is asynchronous.

Question sources

Computer Science MCQ practice collection · Question 121

Question 232 | AM0122

What is long polling in client-server communication?

A. A technique where the server continuously sends updates to the client

B. A technique where the client continuously sends requests to the server for updates

C. A technique where the server holds a request from the client until new data is available

D. A technique where the client holds a request from the server until new data is available

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer C: The server holds a client request until data is available or timeout occurs.

WEB35 defines long polling as holding a client request until an event or timeout, returning a response, and then reconnecting from the client. C captures that defining held-request step. It is not continuous unsolicited streaming on a finished response, repeated short polling without a held request, or a reversed arrangement in which a client holds a server-originated request.

Why each choice is right or wrong

A - Incorrect: Continuous unsolicited server updates describe streaming-like behavior rather than the held-request long-polling cycle.

B - Incorrect: Repeated immediate polling requests describe short polling rather than holding each request until an event or timeout.

C - Correct: Holding the client's request until new data or timeout is the defining long-polling behavior.

D - Incorrect: This reverses the roles; the server holds a request initiated by the client.

Primary note proof

Study-note section WEB35

> Short polling repeatedly requests updates.

Study-note section WEB35

> Long polling holds one client request until an event or timeout, returns a response, then the client reconnects; it is not continuous unsolicited streaming on that same finished response.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 25-31

> In order to make an HTTP request to the server with JavaScript, you need an instance of an XMLHttpRequest Object. We make a request to the sever, by calling the open() and send() methods of the HTTP request object, like this:
> httpRequest.open('GET', 'http://www.example.org/some.file', true);

httpRequest.send();
open() takes three parameters, the first is the HTTP request method and the second is the URL. The optional third parameter sets whether the request is asynchronous.

Web Programming module | PDF page 90 | lines 2-6

6.4 Handling Response from Server
Before making a request to the server we have to write a callback function which will handle the response. But Because Ajax calls are asynchronous, we can't be sure when the response will come, so we must write code to wait for the response and handle it when it arrives. That is what the callback function is used for

Question sources

Computer Science MCQ practice collection · Question 122

Question 233 | AM0124

Which programming language is commonly used for client-side scripting in web programming?

A. HTML

B. CSS

C. JavaScript

D. PHP

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: JavaScript

The JavaScript chapter identifies the browser behavior/scripting layer as JavaScript. HTML provides structure, CSS appearance and PHP server-side output. Evidence: PDF p. 44, PDF p. 60.

Why each choice is right or wrong

A - Incorrect: Markup rather than scripting.

B - Incorrect: Stylesheet rather than scripting.

C - Correct: Exact client scripting language.

D - Incorrect: Explicit server-side language.

Primary note proof

Study-note section WEB01

HTML supplies structure/meaning; CSS controls presentation; JavaScript supplies programmable interactivity.

Study-note section WEB01

PHP, Python, JSP/Java and ASP.NET applications generate server responses and access databases; C# is commonly used with ASP.NET.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 9-15

JavaScript is a scripting or programming language that allows us to implement complex features on web pages. JavaScript will enable us to display timely content updates, interactive maps, 2D/3D animated graphics etc. JavaScript is one of the three layers of web technologies which lets to add functionality to a web page. The three layers of web technologies are stacked together nicely, HTML will let's structure the webpage, CSS deals with the presentation of the web page and JavaScript the functionality.'

Web Programming module | PDF page 60 | lines 4-10

PHP is a server-side scripting language, which can be embedded in HTML or used as a standalone binary (although the former use is much more common). PHP stands for Hypertext Preprocessor. The product was originally named Personal Home Page Tools, and many people still think that is what the acronym stands for. But as it expanded in scope, a new and more appropriate name was selected by community vote. PHP is the language that you use to make the server generate dynamic output-output that is potentially different each time a browser re-quests a page.

Question sources

Computer Science MCQ practice collection · Question 124

Question 234 | AM0125

Which protocol is commonly used for secure communication between a client and server?

A. HTTP

B. FTP

C. TCP/IP

D. HTTPS

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer D: HTTPS

WEB05 defines HTTPS as HTTP over TLS with channel confidentiality, integrity, and certificate-based server authentication. It is the offered protocol for a protected web client/server exchange. Plain HTTP and the TCP/IP networking family do not supply those protections by themselves, and FTP is not the secure HTTP protocol being described. These protections do not replace application validation or authorization.

Why each choice is right or wrong

A - Incorrect: Plain HTTP lacks the TLS protections that distinguish secure web communication in WEB05.

B - Incorrect: FTP is not the HTTPS protocol identified for TLS-protected web exchanges.

C - Incorrect: TCP/IP supplies underlying networking rather than the specific protected HTTP channel requested.

D - Correct: HTTPS adds TLS confidentiality, integrity, and checked server identity to HTTP communication.

Primary note proof

Study-note section WEB05

> HTTPS uses HTTP over TLS, protecting channel confidentiality/integrity and authenticating the server under certificate checks.

Study-note section WEB-S14

> HTTP defines web application requests, response status codes and representations.

Study-note section WEB-S14

> An HTTP exchange can use underlying TCP connections and IP addresses; naming that network stack does not identify the application-level method or status semantics.

Study-note section WEB-S14

> FTP is a file-transfer application protocol.

Study-note section WEB-S14

> SMTP transports electronic mail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 58 | lines 4-8

> window.location.href returns the href (URL) of the current page
> window.location.hostname returns the domain name of the web host
> window.location.pathname returns the path and filename of the current page
> window.location.protocol returns the web protocol used (http: or https:)
> window.location.assign() loads a new document

Web Programming module | PDF page 8 | lines 29-34

• The World Wide Web (WWW) is most often called the Web.
• The Web is a network of computers all over the world.
• All the computers in the Web can communicate with each other.
• All the computers use a communication standard called HTTP.
How does the WWW work?
• Web information is stored in documents called Web pages.

Question sources

Computer Science MCQ practice collection · Question 125

Question 235 | AM0126

What is a load balancer in the context of client-server systems?

A. A device that balances the load between multiple clients

B. A device that balances the load between multiple servers

C. A device that balances the load between the client and server

D. A device that balances the load between different network protocols

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer B: Distribute work across multiple server instances.

WEB36 says workers or processes and a load balancer distribute work across server instances. That directly supports B. The balancing target is the pool of servers handling incoming work, rather than clients, the two communication roles, or protocol types. The note also warns that distributing requests does not automatically coordinate shared data or session state.

Why each choice is right or wrong

A - Incorrect: The described load-balancing role distributes server work rather than balancing computational load among clients.

B - Correct: A load balancer distributes incoming work across multiple server instances.

C - Incorrect: It does not divide one application’s work equally between the abstract client and server roles.

D - Incorrect: Balancing among network protocol types is not the server-instance distribution role defined in WEB36.

Primary note proof

Study-note section WEB36

Workers/processes and a load balancer distribute work across server instances; they do not automatically coordinate shared data or session state.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 9 | lines 13-21

> Web hosting is the place where all the files of your website live. Web hosting is a service that allows organizations and individuals to post a website or a web page onto the Internet. A web host, or web hosting service provider, is a business that provides the technologies and services needed for the website or webpage to be viewed in the Internet. Websites are hosted, or stored, on special computers called servers. When Internet users want to view your website, all they need to do is type your website address or domain (the domain name for the site has to be registered) into their browser. Their computer will then connect to your server and your webpages will be delivered to them through the browser.

Web Programming module | PDF page 92 | lines 6-14

> b. Node.js
> Node is an open source, cross platform runtime environment that allows developers to create server-side tools and applications in JavaScript. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.
> Benefits of using Node
> 1. Performance
> 2. Uses plain old JavaScript
> 3. Thousands of reusable packages
> 4. Portable

Question sources

Computer Science MCQ practice collection · Question 126

Question 236 | AM0127

Which browser-stored mechanism normally carries an opaque session identifier automatically on matching requests, without adding that token to each URL?

A. Cookies

B. HTML forms

C. URL parameters

D. Server-side caching

Hint

Follow the identifier and server-side state through the stated lifecycle or access check; distinguish client-controlled claims from server decisions.

Answer A: Cookies

A cookie stores browser-side name/value data and is automatically attached to requests that match its scope and security rules. It can carry the opaque identifier used to find server-side session state. Forms or URL parameters can explicitly transmit identifiers, but do not provide the automatic matching-request behavior specified. Server-side caching stores server data rather than supplying this browser identifier carrier.

Why each choice is right or wrong

A - Correct: A matching cookie is automatically attached to requests and can carry the opaque session identifier.

B - Incorrect: A form submits explicit values when submitted, rather than automatically attaching stored data to all matching requests.

C - Incorrect: A URL parameter must be included in a URL; the scenario explicitly keeps the token out of URLs.

D - Incorrect: Server caching concerns retained server-side data, not the browser mechanism that automatically carries the identifier.

Primary note proof

Study-note section WEB28

> Cookies are browser-held name/value data automatically attached to matching requests subject to domain, path, expiry, Secure and SameSite rules.

Study-note section WEB29

> Normally the browser holds an opaque session identifier while data resides in server-controlled storage.

Study-note section WEB29

> Protect session IDs, regenerate after login, expire appropriately and avoid putting sensitive tokens in URLs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 58 | lines 14-23

> Cookies are data, stored in small text files, on your computer.
> When a web server has sent a web page to a browser, the connection is shut down,
> and the server forgets everything about the user. Cookies were invented to solve the
> problem "how to remember information about the user":
> When a user visits a web page, his/her name can be stored in a cookie.
> Next time the user visits the page, the cookie "remembers" his/her name.
> Cookies are saved in name-value pairs like:
> When a browser requests a web page from a server, cookies belonging to the page
> are added to the request. This way the server gets the necessary data to "remember"
> information about users.

Web Programming module | PDF page 77 | lines 17-21

> What is a Cookie?
> A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.

Web Programming module | PDF page 76 | lines 2-5

> in one application. A PHP session allows you to store user information on the server for later use (i.e., username, shopping items, etc.). However, session information is temporary and will be deleted after the user has left the website. If you need a permanent storage you may want to store the data in a database.

Question sources

Computer Science MCQ practice collection · Question 127

Question 237 | AM0128

What is AJAX in web programming?

A. A programming language for server-side scripting

B. A database management system

C. A technique for asynchronous client-server communication

D. A web server software

Hint

Identify the execution side and communication pattern, keeping runtime, transport, data format and callback roles distinct.

Answer C: A technique for asynchronous client-server communication

AJAX is a web-development technique with asynchronous server communication and page updates without refresh. It combines technologies instead of being a server language, database system or server software. Evidence: PDF p. 89.

Why each choice is right or wrong

A - Incorrect: Not the module classification.

B - Incorrect: Not the module classification.

C - Correct: Exact asynchronous communication technique.

D - Incorrect: Not the module classification.

Primary note proof

Study-note section WEB34

> AJAX means Asynchronous JavaScript and XML, but responses can be JSON, text, HTML or XML.

Study-note section WEB34

It is a technique, not a language, standalone framework or mandatory XML protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 89 | lines 4-13

AJAX is a web development technique for creating interactive web applications.
AJAX stands for Asynchronous JavaScript and XML. It can send and receive information in various formats, including JSON, XML, HTML, and text files. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script. AJAX's most appealing characteristic is its "asynchronous" nature, which means it can communicate with the server, exchange data, and update the page without having to refresh the page.
The two major features of AJAX allow you to do the following:
- Make requests to the server without reloading the page
- Receive and work with data from the server

Question sources

Computer Science MCQ practice collection · Question 128

Question 238 | AM0129

What is a RESTful API?

A. An API that supports real-time communication between client and server

B. An API that uses the REST architectural style for client-server communication

C. An API that only supports server-to-server communication

D. An API that is used for database management

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer B: An API following the REST architectural style.

WEB05 describes REST through client/server separation, stateless requests, cacheability, and a uniform resource interface. B is the definition matching those architectural constraints. Real-time delivery is not required, server-to-server-only use is not a REST limitation, and database management is only a possible implementation domain. Returning JSON alone also does not establish REST.

Why each choice is right or wrong

A - Incorrect: Real-time communication is not the defining constraint or requirement of a RESTful API.

B - Correct: Following REST architectural constraints is what makes an API RESTful.

C - Incorrect: REST is not restricted to server-to-server communication; browsers can access resource interfaces too.

D - Incorrect: Database management is one possible domain, not the architectural definition of REST.

Primary note proof

Study-note section WEB05

> A RESTful API follows constraints such as client/server separation, stateless requests, cacheability and a uniform resource interface.

Study-note section WEB05

> Resources have identifiers and representations; JSON is common but not required.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 44 | lines 21-28

> We can use JavaScript in coordination with Application programming interfaces (APIs). APIs provide us extra information. They are ready made code building blocks that are used to simplify programming in JavaScript. There are two types of APIs
> - Third party API → are not built into the browser. We have to grab the code from the web. Examples: Twitter API, Google maps API etc.
> - Browser API → are built into the browser. Examples DOM (Document object model) which is covered later in this chapter, Geolocation API, Canvas and WebGL, Audio and Video elements

Web Programming module | PDF page 89 | lines 20-24

> 6.3 Sending Request to PHP server
> XMLHttpRequest (XHR) is an API that can be used by JavaScript and other web browser scripting languages to transfer and manipulate XML data to and from a webserver using HTTP, establishing an independent connection channel between a webpage's Client-Side and Server-Side.

Question sources

Computer Science MCQ practice collection · Question 129

Question 239 | AM0130

What is server-side caching in client-server systems?

A. A technique to store client-side data on the server

B. A technique to store server-side data on the client

C. A technique to store frequently accessed server-side data in memory for faster retrieval

D. A technique to store client and server data in a separate cache server

Hint

Distinguish the purpose of each development or performance technique from its possible side effects.

Answer C: Retain frequently accessed server-side results for faster reuse.

WEB39 describes caching as reusing a representation or result and explicitly gives server caching of rendered pages or expensive query results. C matches that faster-retrieval purpose. The cache need not be a separate server, and merely transferring client data to a server or server data to a client is not the defining reuse mechanism. Validity, privacy, and cache keys still matter.

Why each choice is right or wrong

A - Incorrect: Storing client-originated data on a server does not by itself establish result reuse or caching.

B - Incorrect: Placing server data on a client describes client-side storage rather than server-side caching.

C - Correct: Retaining frequently reused server-side results avoids repeated work and accelerates later retrieval.

D - Incorrect: A separate cache server is optional; the defining property is reusable results, not a mandatory deployment topology.

Primary note proof

Study-note section WEB39

> Caching reuses a representation/result and requires validity/invalidation rules.

Study-note section WEB39

> Server caching can hold rendered pages or expensive query results; never share private user content under an unsafe common key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 45 | lines 28-31

> As a rule, only the simplest scripts are put into HTML. More complex ones reside in separate files.
> The benefit of a separate file is that the browser will download it and store it in its cache.

Web Programming module | PDF page 76 | lines 2-5

> in one application. A PHP session allows you to store user information on the server for later use (i.e., username, shopping items, etc.). However, session information is temporary and will be deleted after the user has left the website. If you need a permanent storage you may want to store the data in a database.

Question sources

Computer Science MCQ practice collection · Question 130

Question 240 | CS0032

Which form METHOD value normally places submitted name/value pairs in the request body rather than the URL query?

A. GET

B. INPUT

C. ACTION

D. POST

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer D: POST

For ordinary HTML forms, GET usually puts successful name/value pairs in the URL query, while POST places form data in the request body. METHOD does not specify an element-at-a-time parser; parsing depends on encoding and the receiving implementation. INPUT is a control element name and ACTION is a form attribute, so neither is a method value. The corrected question tests the supported form-submission distinction rather than the unsupported parsing premise.

Why each choice is right or wrong

A - Incorrect: GET normally places the pairs in the URL query.

B - Incorrect: INPUT names a control element rather than a METHOD value.

C - Incorrect: ACTION identifies a separate form attribute rather than a METHOD value.

D - Correct: POST normally puts submitted form data in the request body.

Primary note proof

Study-note section WEB04

> GET forms usually put name/value pairs in the URL query; POST forms put them in the body.

Study-note section WEB09

> A form’s action identifies its processing URL, method its submission method, and enctype its encoding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 21 | lines 23-31

Form attributes:
- action → Backend script ready to process the data collected
- method → method used to upload Data. The most frequently used
- target → Specify the target window or frame where the result of the script will be displayed. It takes values like _blank, _self, _parent etc.
- enctype → tells how the browser encodes data. Possible values
1. application/x-www-form-urlencoded → most common and used from simple scenarios
2. mutlipart/form-data → used to upload binary data

Web Programming module | PDF page 73 | lines 11-16

• $_GET[]: It is used the retrieve the information from the form control through the parameters sent in the URL. It takes the attribute given in the url as the parameter.
• $_POST[]: It is used the retrieve the information from the form control through the HTTP POST method. IT takes name attribute of corresponding form control as the parameter.

Question sources

Computer Science Exit Exam practice collection · Question 32

Question 241 | CS0033

Which claim is false about HTML as an open-standard text markup format?

A. It supports interoperability, though feature support varies across platforms and browsers

B. Its text can be transmitted over a network as a web representation

C. Its source represents document structure using text markup rather than requiring a fixed small file size

D. Because it is proprietary technology

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer D: Because it is proprietary technology

HTML supplies document structure in a text markup representation within the open Web standards setting. That supports interoperability but does not guarantee identical feature support in every browser, a tiny document size or a fixed transfer speed. The alternatives remove those universal promises. Calling HTML proprietary contradicts the stated open-standard setting, while the other choices now make qualified claims about its representation and delivery.

Why each choice is right or wrong

A - Incorrect: Qualified interoperability is consistent with differing feature support.

B - Incorrect: A web representation can transmit HTML text without promising a fixed speed.

C - Incorrect: Text markup expresses structure without guaranteeing a small size.

D - Correct: The proprietary characterization conflicts with the open-standard setting.

Primary note proof

Study-note section WEB02

> Open standards do not guarantee identical support for every feature in every browser.

Study-note section WEB06

> HTML means HyperText Markup Language.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 10 | lines 17-26

> HTML has been in use by the World-Wide Web (WWW) global information initiative since 1990. It is a simple markup language used to create hypertext documents that are platform independent. HTML markup can represent hypertext news, mail, documentations, and hyper media; menus of options; database query results;
> This document covers html versions ranging from 2 up to 5
> HTML is composed of several tags. HTML Document:-
> - A text document containing markup tags
> - The tags tell the browser how to display the document
> - Should have an .htm or .html file name extension
> - Can be created using a simple text editor

Question sources

Computer Science Exit Exam practice collection · Question 33

Question 242 | CS0034

What is the output of the following JavaScript fragment? for (n = 1; n < 7; n++) { if (n % 5 == 0) { document.write(n + " is multiple of 5"); } }

A. 5 is multiple of 5

B. 1 2 3 4 5 is multiple of 5 6

C. 5 is multiple of 5 6 7

D. 1 2 3 4 5 6

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: 5 is multiple of 5

The loop starts at 1 and visits integers 1 through 6 because n<7 excludes 7. The p.49 arithmetic table shows % as remainder, including 8%3=2. Among these values only 5 has remainder zero on division by 5, so the conditional writes once: 5 is multiple of 5. This uses the conventional n++ loop step; the module shows for begin/condition/step but does not separately define postfix value semantics, which are immaterial because the step value is discarded. Evidence: PDF p. 49, PDF p. 49, PDF p. 50, PDF p. 47.

Why each choice is right or wrong

A - Correct: Exactly the single matching value 5.

B - Incorrect: Prints values outside the if branch; no unconditional write exists.

C - Incorrect: Adds 6 and 7 even though 6 fails and 7 is excluded.

D - Incorrect: Ignores the conditional and omits the literal message.

Primary note proof

Study-note section WEB19

> for(initialization;test;update) initializes once, checks before each iteration, then updates after each body.

Study-note section WEB-S13

> For nonnegative integers n and positive d, n % d gives the remainder after division; zero tests divisibility.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 49 | lines 2-5

> 4.6 Arithmetic and Logical Operators in JavaScript
> Arithmetic operators are used to perform mathematical operations between numeric operands. We can perform different operations on numbers, like addition, subtraction multiplication. Here is the list of arithmetic operators we can use on numbers

Web Programming module | PDF page 49 | lines 9-14

> 4.7 Control Structures (Conditional and Looping Statements)
> The “if” statement:
> The if(...) statement evaluates a condition in parentheses and, if the result is true, executes a block of code.
> let age = 20;
> if (age > 18) alert('You are right!');

Web Programming module | PDF page 50 | lines 30-33

> - For loop → syntax
> for (begin; condition; step) {

```
// ... loop body ...
}
```

Web Programming module | PDF page 47 | lines 3-9

3. The write() method
- The write() method writes HTML expressions or JavaScript code to a document.
- The write() method is mostly used for testing: If it is used after an HTML document is fully loaded, it will delete all existing HTML.
Usage:
document.write(expr)

Question sources

Computer Science Exit Exam practice collection · Question 34

Question 243 | CS0035

In an ordinary browser page, which statements about document.write("Hello World") are true? Classify properties and values inclusively: a method is also a function-valued property. Select all that apply.

A. write is a property

B. write is a method

C. document is a property

D. write is an object

Hint

Track the value and type at each operation, applying the stated conversion, indexing or mathematical rule.

Answer A, B, C, D: write is a property; write is a method; document is a property; write is an object

document.write is a method and also a function-valued property of document. In a normal browser page, document is exposed as a property of window. Functions are objects in JavaScript, so the write function value is also an object. These descriptions are compatible views of the same expression rather than mutually exclusive categories. All four statements are therefore true under the explicit inclusive-classification convention.

Why each choice is right or wrong

A - Correct: write is a function-valued property of document.

B - Correct: Calling that function through document is a method invocation.

C - Correct: document is the browsing context’s document property.

D - Correct: The write function value is an object because JavaScript functions are objects.

Primary note proof

Study-note section WEB22

> In a normal browser page, window exposes the browsing context and document is its document property.

Study-note section WEB22

> document.write is a method/function property; writing after document load can replace the document.

Study-note section WEB16

> JavaScript primitives are undefined, null, Boolean, Number, BigInt, String and Symbol; objects include arrays, functions and dates.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Web Programming module | PDF page 47 | lines 3-9

> 3. The write() method
> - The write() method writes HTML expressions or JavaScript code to a document.
> - The write() method is mostly used for testing: If it is used after an HTML document is fully loaded, it will delete all existing HTML.
> Usage:
> document.write(expr)

Web Programming module | PDF page 56 | lines 17-25

> All global JavaScript objects, functions, and variables automatically become members of the window object.
> Global variables are properties of the window object
> Global functions are methods of the window object.
> Even the document object (of the HTML DOM) is a property of the window object:
> window.document.getElementById("header");
> is the same as:
> document.getElementById("header");

Question sources

Computer Science Exit Exam practice collection · Question 35

Question 244 | NS0284

Consider
nav ul {
list-style-type: none;
margin: 0;
padding: 0;
}
What problem does this solve when creating a navigation bar?

A. Removes bullet points and default spacing, allowing custom layout

B. Aligns the navbar to the right of the page

C. Adds hover effects to list items

D. Converts the list into a dropdown menu

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer A: Remove list bullets and default margin/padding.

This rule resets navigation-list styling. list-style-type:none removes markers, and zero margin/padding removes default list spacing. None of those declarations specifies alignment, hover styling, or dropdown behavior. Thus A describes the actual effect, leaving layout and interaction to additional rules.

Why each choice is right or wrong

A - Correct: These declarations remove markers and default spacing so later layout rules can start from a clean list.

B - Incorrect: No text alignment, flex justification, or positioning rule aligns the navbar right.

C - Incorrect: No :hover selector or hover-specific declaration is present.

D - Incorrect: A dropdown needs additional layout and interaction rules, not just this reset.

Primary note proof

Study-note section WEB08

> Resetting nav ul with list-style-type:none; margin:0; padding:0 removes bullets and default spacing only; it does not itself arrange links horizontally or create a dropdown.

Question sources

Computer Science Exit Exam · 2018 · Question 1

Question 245 | NS0311

Which PHP superglobal is dedicated to URL query parameters, excluding configurable combined-source arrays?

A. $_REQUEST

B. $_POST

C. $_GET

D. isset()

Hint

Identify the data source or security responsibility precisely, and check whether a combined source or a different control could also qualify.

Answer C: $_GET

$_GET is the dedicated query-parameter source. $_REQUEST can also contain GET values depending on configuration, but it is a combined-source array and is explicitly excluded by this question. $_POST represents supported form POST data, while isset tests existence and non-null rather than acting as an input-data container. Removing placeholder brackets makes the array names clear without inventing a particular field name or request value.

Why each choice is right or wrong

A - Incorrect: This combines configured request sources rather than being dedicated to query parameters.

B - Incorrect: This is the supported form POST source.

C - Correct: This is the dedicated URL-query source.

D - Incorrect: This tests whether a variable or entry exists and is non-null.

Primary note proof

Study-note section WEB27

> $_GET contains query parameters; $_POST contains supported form POST data; $_FILES upload metadata; $_SERVER request/environment details; $_COOKIE cookie values; $_SESSION session data after startup.

Study-note section WEB27

> $_REQUEST may combine GET/POST/cookie data according to configuration; it is not a database interface and has ambiguous precedence, so prefer the intended source.

Study-note section WEB27

> isset tests existence and non-null, not truth or valid syntax.

Question sources

Computer Science Exit Exam · 2018 · Question 32

Question 246 | NS0332

_____is a service that allows organizations and individuals to post a website or a web page onto the Internet.

A. Web hosting

B. Internet service provider

C. Web client

D. Domain name registration

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer A: Web hosting

WEB02 defines hosting as supplying the environment where a website is served. This is the service enabling an organization’s site or page to be made available online. An ISP supplies connectivity, domain registration reserves a name, and a web client requests resources. Those supporting roles are not the hosting service described.

Why each choice is right or wrong

A - Correct: Hosting supplies an environment in which the site is served to users.

B - Incorrect: An ISP provides connectivity rather than necessarily the environment serving the website.

C - Incorrect: A web client requests resources instead of providing the publishing environment.

D - Incorrect: Domain registration reserves a name rather than serving the site’s content.

Primary note proof

Study-note section WEB02

> Hosting supplies an environment where a site is served; domain registration reserves a name, and DNS maps names to network records.

Study-note section WEB02

> An ISP provides connectivity.

Question sources

Computer Science Exit Exam · 2018 · Question 59

Question 247 | NS0340

Which protocol is mostly used for communication between client and server in a web application?

A. FTP

B. SMTP

C. HTTP

D. SSH

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer C: HTTP

WEB01 describes the Web as a resource system commonly using HTTP, and WEB03 details its request/response exchange. HTTPS adds TLS protection to HTTP but does not make one of the other offered protocols the ordinary web exchange. Among these choices, HTTP is the application protocol used to request and return web resources.

Why each choice is right or wrong

A - Incorrect: FTP is not the HTTP request/response web protocol described in the note.

B - Incorrect: SMTP is not the webpage request/response protocol developed in WEB03.

C - Correct: HTTP carries the web resource requests and responses described in the complete note.

D - Incorrect: SSH is not the ordinary webpage request/response protocol identified here.

Primary note proof

Study-note section WEB01

> The Internet connects networks; the Web is a resource system commonly using HTTP.

Study-note section WEB03

> An HTTP/1.1 request begins with method, request-target and version, followed by header fields, a blank line, and an optional body.

Study-note section WEB-S14

> HTTP defines web application requests, response status codes and representations.

Study-note section WEB-S14

> An HTTP exchange can use underlying TCP connections and IP addresses; naming that network stack does not identify the application-level method or status semantics.

Study-note section WEB-S14

> FTP is a file-transfer application protocol.

Study-note section WEB-S14

SMTP transports electronic mail.

Question sources

Computer Science Exit Exam · 2018 · Question 71

Question 248 | NS0342

What is the extension of JavaScript file?

A. .javaS

B. .JS

C. .java

D. CSS

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B: .js is the conventional JavaScript source extension.

WEB15 uses an external app.js script and explicitly calls it a .js file. B identifies that extension family despite its uppercase printed letters. Actual filename case must match the deployed resource where paths are case-sensitive; the conventional spelling is lowercase .js. The other choices name different or invented file extensions rather than JavaScript source.

Why each choice is right or wrong

A - Incorrect: The invented .javaS suffix is not the conventional .js extension specified by WEB15.

B - Correct: This identifies the JS extension; use conventional lowercase .js and preserve actual filename case in paths.

C - Incorrect: .java is not the .js source-file extension used for JavaScript in the note.

D - Incorrect: CSS identifies stylesheet source rather than the JavaScript file extension.

Primary note proof

Study-note section WEB15

> Use <script src="app.js"></script> for an external .js file; script content has no HTML closing-tag syntax inside its JavaScript source.

Study-note section WEB15

> JavaScript originated at Netscape; it is distinct from Java and supports sophisticated applications through prototype-based objects and first-class functions.

Question sources

Computer Science Exit Exam · 2018 · Question 73

Question 249 | NS0355

A web application has two protected endpoints. The profile endpoint checks a valid session identifier and its server-side user record on every request. The transfer endpoint grants access whenever the editable logged_in cookie equals 1 and ignores session validation. An unauthenticated attacker has no valid session and sets logged_in=1. Which result follows from these access checks?

A. The cookie edit can bypass the transfer endpoint, but does not grant access to the profile endpoint

B. The cookie edit grants access to both endpoints because both receive the same cookie

C. Neither endpoint can be bypassed unless the attacker first obtains a genuine session identifier

D. HTTPS alone prevents either endpoint from accepting a forged logged_in value

Hint

Follow the identifier and server-side state through the stated lifecycle or access check; distinguish client-controlled claims from server decisions.

Answer A: The cookie edit can bypass the transfer endpoint, but does not grant access to the profile endpoint

The two endpoints use different trust paths. The transfer endpoint trusts an attacker-controlled flag, so the forged value satisfies its stated access check without a valid session. The profile endpoint requires authenticated server-side state, which the attacker lacks, so the unrelated cookie does not grant access there. HTTPS protects transport rather than validating the truth of a client-generated login claim. The bypass therefore affects the transfer endpoint only.

Why each choice is right or wrong

A - Correct: Only the transfer endpoint relies solely on the forgeable flag; the profile endpoint still requires a valid session.

B - Incorrect: Receiving the same cookie does not make the endpoints apply the same authorization rule.

C - Incorrect: The transfer check requires no genuine session identifier, so stealing one is unnecessary for that bypass.

D - Incorrect: HTTPS does not prevent a client from submitting a false authorization flag.

Primary note proof

Study-note section WEB29

> A client cookie logged_in=1 is attacker-controlled.

Study-note section WEB29

> If the server trusts it alone, authorization can be bypassed.

Study-note section WEB29

If the server checks a valid session ID and its server-side user/role on every protected action, editing that unrelated flag does not grant rights.

Study-note section WEB29

HTTPS does not make client-generated claims trustworthy.

Question sources

Computer Science Exit Exam · 2018 · Question 93

Question 250 | NS0360

Why should form <input> elements include an associated <label> with the for attribute?

A. It enables offline form storage

B. It encrypts user data

C. It increases form submission speed

D. It improves accessibility: screen readers announce the label when the input is focused

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer D: Provide an accessible label associated with the input.

WEB09 links a label's for to the control id, and WEB07 requires programmatic control labels. That association lets assistive technology identify the input and announce its label. It does not alter storage, encryption, or network submission speed. D states the accessibility purpose directly, rather than confusing it with unrelated browser or transport features.

Why each choice is right or wrong

A - Incorrect: Label association does not create offline storage for form values.

B - Incorrect: Labels do not encrypt data or replace HTTPS transport protection.

C - Incorrect: Associating a label does not accelerate form submission over the network.

D - Correct: The programmatic label identifies the focused input for screen readers and other assistive technology.

Primary note proof

Study-note section WEB09

> A label's for matches the control's id; name identifies its submitted field.

Study-note section WEB07

> Give controls programmatic labels, clear link text and keyboard access; visual styling alone does not establish accessibility.

Question sources

Computer Science Exit Exam · 2018 · Question 99

Question 251 | NS0795

What is the difference between GET and POST methods? When would you use each?

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer: GET retrieves safely and usually places form values in the query; POST asks the target to process body content.

WEB04 recommends GET for reading representations rather than intentional state changes. POST suits submitted processing or actions such as enrollment with proper authorization and CSRF controls. Neither method encrypts data; use HTTPS for channel protection. POST body values remain visible to tools and subject to configured size limits rather than becoming secret or unlimited.

Primary note proof

Study-note section WEB04

> GET retrieves a representation with safe semantics; do not implement an intentional state-changing action as a GET link.

Study-note section WEB04

> POST asks the target to process supplied content.

Study-note section WEB04

> GET forms usually put name/value pairs in the URL query; POST forms put them in the body.

Study-note section WEB40

> For an enrollment action use POST with CSRF controls, server authorization, capacity checks and a transaction; show success only after commit.

Question sources

Web Programming review questions · Question 1

Question 252 | NS0796

Why is client-side validation alone not enough for security?

Hint

Separate event dispatch, handler registration and cancellation of the browser default action.

Answer: Clients can bypass or alter browser checks, so the server must validate and authorize again.

WEB27 requires server checks for required fields, type, length, and range, plus authentication and authorization before processing. Browser validation improves feedback but arbitrary clients can submit different requests. Bind SQL values and encode output for its context as well; encryption or merely connecting to a database does not replace these application controls.

Primary note proof

Study-note section WEB27

> Read input, check required fields/type/length/range, authenticate and authorize, then process.

Study-note section WEB27

> Bind SQL parameters to keep values separate from commands.

Study-note section WEB27

> Encode output for HTML/attribute/JS context; textContent handles plain browser text.

Study-note section WEB21

> Browser checks improve feedback; repeat validation and authorization on the server.

Question sources

Web Programming review questions · Question 2

Question 253 | NS0797

What is the DOM and why is it important for web development?

Hint

Track which node is selected or created, and distinguish changing content from attaching a node.

Answer: The DOM is the document's node hierarchy, enabling scripts to inspect and update page content and behavior.

WEB20 selects nodes, creates elements, and attaches them to that hierarchy. Creating a node does not itself insert it, and missing lookups must be guarded. textContent safely inserts untrusted plain text, whereas innerHTML reparses markup and can replace descendant state. The DOM connects document structure to interactive browser updates.

Primary note proof

Study-note section WEB20

> The DOM is a hierarchy of nodes representing the document.

Study-note section WEB20

> Creation alone does not insert it.

Study-note section WEB20

> innerHTML parses/replaces descendants, so += reparses/replaces the prior descendant content too and can destroy listeners/state.

Study-note section WEB20

> Use textContent for untrusted plain text.

Question sources

Web Programming review questions · Question 3

Question 254 | NS0798

A user closes their browser and reopens it. They are still logged in. How is this possible?

Hint

Follow the identifier and server-side state through the stated lifecycle or access check; distinguish client-controlled claims from server decisions.

Answer: A persistent remember/session token or browser restoration can preserve the login flow.

WEB28 says reopening while logged in may use persistent session or remember tokens or browser restoration. The browser can retain an identifier, but the server still has to accept and validate it against its authorization state. Local persistence alone is not proof of authenticated access, and the observation does not identify one unique storage mechanism.

Primary note proof

Study-note section WEB28

> Being logged in after reopening may use a persistent session/remember token or browser restoration; storage alone does not prove server acceptance.

Study-note section WEB29

> If the server checks a valid session ID and its server-side user/role on every protected action, editing that unrelated flag does not grant rights.

Question sources

Web Programming review questions · Question 4

Question 255 | NS0799

Why would a developer choose local storage over cookies?

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer: localStorage persists origin data without automatically attaching it to every matching request.

WEB28 contrasts localStorage's persistent strings with cookies' automatic request transmission. It can avoid transmitting data needed only by the client. JavaScript can still send stored tokens explicitly, so localStorage is not inherently incapable of session designs. Script-access risks and persistence policies must be considered; neither mechanism is universally preferable for sensitive authentication data.

Primary note proof

Study-note section WEB28

> localStorage persists origin-specific string values across sessions until removed/cleared; sessionStorage is scoped to a page session/tab.

Study-note section WEB28

> Neither is automatically transmitted like matching cookies, but JavaScript can send stored data explicitly.

Study-note section WEB28

> Thus all four are client storage techniques, and localStorage can participate in token-based session designs, with script-access risks.

Question sources

Web Programming review questions · Question 5

Question 256 | NS0800

What does HTML stand for?

A. Hyper Text Markup Language

B. High Tech Modern Language

C. Hyper Transfer Markup Language

D. Home Tool Markup Language

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer A: HyperText Markup Language

WEB06 directly expands HTML as HyperText Markup Language. The name identifies markup used for document structure and meaning rather than a general programming or transfer protocol. The other phrases substitute different words and are not the expansion in the note. This vocabulary answer does not depend on a browser version or an implementation choice.

Why each choice is right or wrong

A - Correct: Hyper Text Markup Language matches the note's expansion, with only spacing variation in HyperText.

B - Incorrect: High Tech Modern Language is not the expansion given in WEB06.

C - Incorrect: Transfer replaces the required word Text and gives the wrong expansion.

D - Incorrect: Home Tool Markup Language is not the HTML expansion in the note.

Primary note proof

Study-note section WEB06

> HTML means HyperText Markup Language.

Question sources

Web Programming review questions · Question 1

Question 257 | NS0801

Which HTML tag creates a hyperlink?

A. <link>

B. <href>

C. <a>

D. <url>

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer C: <a> with an href destination.

WEB07 gives Read as the link pattern. The anchor element creates the user-facing hyperlink; href is its destination attribute, not a standalone tag. link is used for relationships such as external stylesheets, and url is not the anchor element. Thus C names the correct element among the supplied choices.

Why each choice is right or wrong

A - Incorrect: link is used for resource relationships such as external stylesheets rather than the shown user-facing anchor.

B - Incorrect: href is an attribute on an anchor, not the hyperlink element itself.

C - Correct: The anchor element with href creates the hyperlink described in WEB07.

D - Incorrect: url is not the supported HTML anchor tag.

Primary note proof

Study-note section WEB07

> Use Read for a link; href is the destination, not url/link.

Study-note section WEB11

> An external sheet uses <link rel="stylesheet" href="styles.css"> in head; an internal sheet uses style; an inline declaration uses an element’s style attribute.

Question sources

Web Programming review questions · Question 2

Question 258 | NS0802

What is the correct HTML tag for inserting a line break?

A. <break>

B. <lb>

C.

D. <newline>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C:

 creates a line break, <p> defines a paragraph, and <hr> represents a thematic break. C is the defined break element; the other options invent element names not defined as HTML elements. A line break is an HTML structure operation rather than a requirement to insert a physical newline into the source file.

Why each choice is right or wrong

A - Incorrect: break is not the HTML line-break element defined in WEB06.

B - Incorrect: lb is not the supported line-break element spelling.

C - Correct: br is explicitly the HTML line-break element.

D - Incorrect: newline is not the HTML line-break element defined in the note.

Primary note proof

Study-note section WEB06

> <p> marks paragraphs;
 is a line break and <hr> a thematic break.

Question sources

Web Programming review questions · Question 3

Question 259 | NS0803

Which tag preserves whitespace and line breaks exactly as written?

A. <p>

B. <pre>

C. <code>

D.

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer B: <pre>

WEB06 gives <pre> the whitespace-preservation role. It is therefore the appropriate element for text whose source spaces and line breaks should be retained. code marks code semantics but does not itself guarantee preformatted whitespace, while paragraph and emphasis elements have other meanings. Custom CSS can alter defaults, but the question concerns the native tag role.

Why each choice is right or wrong

A - Incorrect: A paragraph does not itself preserve all source spacing and line breaks.

B - Correct: pre is the element explicitly defined to preserve source whitespace.

C - Incorrect: code marks code semantics; preformatted whitespace preservation is a separate role.

D - Incorrect: em expresses stress emphasis rather than preformatted source layout.

Primary note proof

Study-note section WEB06

> <pre> preserves source whitespace; <code> marks code.

Study-note section WEB06

> indicates importance and stress; <b>/<i> have other text semantics and are not identical to importance/emphasis.

Question sources

Web Programming review questions · Question 4

Question 260 | NS0804

What is the highest-level heading tag in HTML?

A. <h6>

B. <heading>

C. <h0>

D. <h1>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer D: <h1>

WEB06 defines heading levels h1 through h6 as a hierarchy. h1 is the highest level and h6 the lowest, regardless of CSS choices that can change their visual size. heading and h0 are not the defined heading-level tags. The answer is semantic rank, not whichever text happens to be styled largest.

Why each choice is right or wrong

A - Incorrect: h6 is the lowest of the defined heading levels, not the highest.

B - Incorrect: heading is not one of the defined h1-h6 heading-level tags.

C - Incorrect: h0 is outside the defined heading levels in WEB06.

D - Correct: h1 is the highest semantic heading level in the h1-h6 hierarchy.

Primary note proof

Study-note section WEB06

> Heading levels h1-h6 express hierarchy, with h6 conventionally smallest in default styling, not a mandatory CSS size.

Question sources

Web Programming review questions · Question 5

Question 261 | NS0805

Which element is the root element of an HTML page?

A. <head>

B. <body>

C. <html>

D. <root>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C: <html>

WEB06 explicitly identifies html as the document root, with head holding metadata and body containing document content. Those two elements are inside the document rather than replacing its root. root is not the HTML tag used for this role. Thus the root-element question directly selects C.

Why each choice is right or wrong

A - Incorrect: head holds metadata and is not the document root.

B - Incorrect: body holds document content but is not the document's root element.

C - Correct: html is explicitly defined as the document root in WEB06.

D - Incorrect: root is not the HTML root tag given here.

Primary note proof

Study-note section WEB06

> <!DOCTYPE html> selects modern standards mode; <html> is the document root, <head> holds metadata, and <body> contains document content.

Question sources

Web Programming review questions · Question 6

Question 262 | NS0806

Which attribute provides alternative text for an image?

A. title

B. src

C. alt

D. caption

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C: alt

WEB07 gives img a src for its resource and alt for alternative text. Informative images need meaningful alternatives; decorative ones use empty alt. A title is not a replacement for alt, and a caption is separate visible context rather than the requested attribute. Therefore C names the accessibility text attribute.

Why each choice is right or wrong

A - Incorrect: title is not a replacement for the image’s alternative-text attribute.

B - Incorrect: src identifies the image resource rather than its alternative text.

C - Correct: alt supplies the image’s alternative text, or an empty value for purely decorative images.

D - Incorrect: caption is not the requested image alternative-text attribute.

Primary note proof

Study-note section WEB07

> <img src="photo.jpg" alt="Students in a lab"> supplies source and alternative text.

Study-note section WEB07

> Use empty alt for purely decorative images; informative images need meaningful alternatives.

Study-note section WEB07

> Width/height reserve dimensions; title is not a replacement for alt.

Question sources

Web Programming review questions · Question 7

Question 263 | NS0807

What does target="_blank" do in a link?

A. Opens the link in the same window

B. Opens the link in a new tab or window

C. Closes the current window

D. Opens the link in a parent frame

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer B: Open a new browsing context, commonly a new tab or window.

WEB07 maps _blank to a new context, _self to current, and _parent to parent. Browser preferences decide tab versus window, so B correctly allows either. The target does not close the current window or select the parent frame. This is a browsing-context request rather than a guaranteed window-management UI behavior.

Why each choice is right or wrong

A - Incorrect: The current context is selected by _self rather than _blank.

B - Correct: _blank requests a new context, with tab versus window determined by browser behavior.

C - Incorrect: The target does not request that the current window be closed.

D - Incorrect: The parent context is selected by _parent rather than _blank.

Primary note proof

Study-note section WEB07

> target="_self" uses the current context, _blank a new context, _parent its parent and _top the top-level context.

Study-note section WEB07

> Browser preferences determine tab versus window.

Question sources

Web Programming review questions · Question 8

Question 264 | NS0808

Which HTML tag is used for an unordered list?

A.

B.

C.

D. <list>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C:

WEB08 defines ul as unordered and ol as ordered, with li items inside either. C therefore supplies the requested container. li is only an item, and list is not the defined HTML list-container name. Presentation of markers can be changed with CSS without changing whether the element is the unordered-list structure.

Why each choice is right or wrong

A - Incorrect: ol is an ordered-list container, not the unordered-list container requested.

B - Incorrect: li marks an item inside a list rather than the list container itself.

C - Correct: ul is explicitly the unordered-list container in WEB08.

D - Incorrect: list is not the unordered-list element spelling defined here.

Primary note proof

Study-note section WEB08

> Unordered <ul> and ordered <ol> contain <li> items.

Question sources

Web Programming review questions · Question 9

Question 265 | NS0809

What HTML tag defines a table row?

A. <td>

B. <th>

C. <table>

D. <tr>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer D: <tr>

WEB08 distinguishes the table element, row elements, and header/data cells. tr defines one row, while th and td are cells placed within rows. table contains the complete tabular structure. Therefore D is the row tag, independent of how many columns or spans the cells in that row occupy.

Why each choice is right or wrong

A - Incorrect: td defines a data cell, not the containing row.

B - Incorrect: th defines a header cell, not the containing row.

C - Incorrect: table defines the whole table rather than one row.

D - Correct: tr is the table-row element explicitly identified in WEB08.

Primary note proof

Study-note section WEB08

> <table> contains rows <tr>, header cells <th> and data cells <td>; a td is not the table itself.

Question sources

Web Programming review questions · Question 10

Question 266 | NS0810

What does the colspan attribute do in a table?

A. Sets the column color

B. Merges cells across multiple columns

C. Sets the column width

D. Merges cells across multiple rows

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer B: Make one cell span multiple columns.

WEB08 explicitly distinguishes colspan from rowspan. colspan allocates a cell across columns, often described informally as merging cells horizontally; it does not change a column's color or width rule. rowspan is the vertical counterpart across rows. B therefore matches, while actual placement must still account for positions already occupied by spans.

Why each choice is right or wrong

A - Incorrect: Column color is a styling concern rather than the colspan structural span.

B - Correct: colspan makes one cell occupy positions across multiple columns.

C - Incorrect: Column width is not the meaning of the colspan count.

D - Incorrect: Spanning rows is rowspan, not colspan.

Primary note proof

Study-note section WEB08

> colspan spans columns; rowspan spans rows.

Study-note section WEB08

> Account for occupied cells before placing the next cell, and use header associations for accessible data.

Question sources

Web Programming review questions · Question 11

Question 267 | NS0811

Which meta tag is used for responsive design on mobile devices?

A. <meta name="keywords">

B. <meta name="robots">

C. <meta name="description">

D. <meta name="viewport">

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer D: <meta name="viewport">

WEB06 gives viewport metadata control over mobile viewport behavior, and WEB14 includes it alongside flexible layout and media queries for responsive design. Keywords and description concern metadata for content/discoverability, while robots concerns crawler directives. The viewport entry is the relevant mobile-layout metadata; it does not replace responsive CSS by itself.

Why each choice is right or wrong

A - Incorrect: Keywords metadata does not configure the mobile viewport used for responsive layout.

B - Incorrect: Robots metadata concerns crawler handling rather than the mobile viewport.

C - Incorrect: Description summarizes the page rather than controlling mobile viewport behavior.

D - Correct: Viewport metadata configures mobile viewport behavior as part of responsive design.

Primary note proof

Study-note section WEB06

> Meta charset declares encoding; description summarizes the page; viewport controls mobile viewport behavior.

Study-note section WEB14

> Responsive design combines flexible layout, images, viewport metadata and media queries, for example @media(max-width:600px){.cards{grid-template-columns:1fr}}.

Study-note section WEB-S12

> The keywords metadata name supplies terms relevant to the page, while description supplies a page summary.

Study-note section WEB-S12

> Robots metadata supplies crawler indexing or search-presentation instructions.

Question sources

Web Programming review questions · Question 12

Question 268 | NS0812

Which CSS selector targets an element with id="logo"?

A. .logo

B. logo

C. *logo

D. #logo

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer D: #logo

WEB11 explicitly uses #logo as an ID selector and .notice as a class selector. Thus an element whose id is logo is targeted with #logo. A bare logo would be a type selector for an element named logo, while the starred concatenation is not the stated ID syntax. D is the correct selector.

Why each choice is right or wrong

A - Incorrect: .logo targets a class named logo rather than id="logo".

B - Incorrect: Bare logo is a type selector, not the ID selector requested.

C - Incorrect: *logo is not the defined CSS ID-selector syntax.

D - Correct: #logo is the explicit ID-selector pattern supplied in WEB11.

Primary note proof

Study-note section WEB11

> Type p selects paragraph elements, .notice selects a class, #logo an ID, h1,h2 groups selectors, nav ul selects descendants and nav>ul direct children.

Question sources

Web Programming review questions · Question 13

Question 269 | NS0813

Which CSS selector targets all elements with class="highlight"?

A. #highlight

B. .highlight

C. *highlight

D. highlight

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer B: .highlight

WEB11 gives a dot-prefixed name as a class selector and a hash-prefixed name as an ID selector. Applying that rule to highlight yields .highlight. A bare word selects element type, and *highlight is not the supported class syntax. The class rule can match every element carrying that class, rather than only one unique ID.

Why each choice is right or wrong

A - Incorrect: #highlight selects an ID rather than elements with the highlight class.

B - Correct: .highlight is the dot-prefixed class selector matching the requested class.

C - Incorrect: *highlight is not the defined class-selector syntax.

D - Incorrect: Bare highlight is an element-type selector rather than a class selector.

Primary note proof

Study-note section WEB11

> Type p selects paragraph elements, .notice selects a class, #logo an ID, h1,h2 groups selectors, nav ul selects descendants and nav>ul direct children.

Question sources

Web Programming review questions · Question 14

Question 270 | NS0814

Which method of inserting CSS allows it to apply to multiple pages?

A. Inline

B. Internal

C. External

D. Embedded

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer C: External stylesheet

WEB11 says external CSS promotes reuse and consistent design through a linked stylesheet. Multiple documents can link the same styles.css file, so one shared source applies across pages. Inline declarations belong to one element and internal/embedded style blocks belong to their document. They can be copied, but copying is not the shared-source attachment method requested.

Why each choice is right or wrong

A - Incorrect: Inline declarations belong to an element rather than one shared stylesheet linked by multiple pages.

B - Incorrect: An internal style block belongs to its document rather than a separately shared resource.

C - Correct: Multiple pages can link the same external stylesheet for shared rules and maintenance.

D - Incorrect: Embedded styles are document-local rather than the shared external attachment method.

Primary note proof

Study-note section WEB11

> An external sheet uses <link rel="stylesheet" href="styles.css"> in head; an internal sheet uses style; an inline declaration uses an element’s style attribute.

Study-note section WEB11

> External CSS promotes reuse, consistent design and caching; it does not guarantee that every page becomes smaller or accessible.

Question sources

Web Programming review questions · Question 15

Question 271 | NS0815

What symbol is used to group multiple selectors in CSS?

A. Semicolon (;)

B. Colon (:)

C. Comma (,)

D. Dot (.)

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer C: Comma

WEB11 groups selectors using h1,h2. The comma separates alternative selectors that receive the same declaration block. Semicolons separate declarations, colons separate a property from its value or occur in pseudo syntax, and dots introduce classes. Thus C is the selector-list grouping separator, not a declaration delimiter.

Why each choice is right or wrong

A - Incorrect: Semicolons separate declarations rather than selectors in a selector list.

B - Incorrect: A colon is not the separator for grouped alternative selectors.

C - Correct: A comma groups selector alternatives, as in the note's h1,h2 example.

D - Incorrect: A dot introduces a class selector rather than grouping multiple selectors.

Primary note proof

Study-note section WEB11

> Type p selects paragraph elements, .notice selects a class, #logo an ID, h1,h2 groups selectors, nav ul selects descendants and nav>ul direct children.

Study-note section WEB11

> A rule is selector { property: value; }.

Question sources

Web Programming review questions · Question 16

Question 272 | NS0816

What does text-decoration: underline do in CSS?

A. Makes text bold

B. Makes text italic

C. Underlines text

D. Strikes through text

Hint

Identify the requested presentation effect and layout context before matching the property or rule.

Answer C: Underline the text.

WEB13 assigns text-decoration effects such as underline. The specific value underline therefore adds an underline decoration. Bold uses font-weight and italic uses font-style; a strike-through is a different decoration value. The declaration changes presentation rather than the semantic importance or stress of the underlying HTML text.

Why each choice is right or wrong

A - Incorrect: Boldness is controlled by font-weight rather than the underline decoration.

B - Incorrect: Italic styling uses font-style rather than text-decoration:underline.

C - Correct: The underline value requests an underline text decoration.

D - Incorrect: A strike-through is a different decoration, not the underline value shown.

Primary note proof

Study-note section WEB13

> color sets text color; background-color sets background; font-size size, font-family face, font-style italic, font-weight weight, text-align inline-content alignment, and text-decoration effects such as underline.

Question sources

Web Programming review questions · Question 17

Question 273 | NS0817

Which HTML tag makes text bold with semantic importance?

A. <b>

B. <bold>

C.

D.

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C:

WEB06 says strong expresses importance while em expresses stress, and warns that b/i are not identical to those meanings. Thus the semantic-importance requirement selects strong rather than merely a bold-looking b. Default styling is commonly bold but CSS can change appearance; the semantic distinction remains the reason for the answer.

Why each choice is right or wrong

A - Incorrect: b has different text semantics and is not the element for importance specified in WEB06.

B - Incorrect: bold is not the semantic importance element defined in the note.

C - Correct: strong explicitly indicates importance and is the requested semantic choice.

D - Incorrect: em marks stress emphasis rather than the stated importance role.

Primary note proof

Study-note section WEB06

> indicates importance and stress; <b>/<i> have other text semantics and are not identical to importance/emphasis.

Question sources

Web Programming review questions · Question 18

Question 274 | NS0818

Which HTML element is used for superscript text like E=mc²?

A. <sub>

B. <sup>

C. <small>

D.

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer B: <sup>

WEB06 pairs sub and sup with subscript and superscript. The raised exponent in E=mc² is a superscript, so sup is the matching element. sub would lower text, and small or em have other meanings rather than expressing the exponent's raised placement. The answer identifies the markup role rather than a CSS font-size choice.

Why each choice is right or wrong

A - Incorrect: sub marks subscript, the opposite vertical placement from the raised exponent.

B - Correct: sup marks superscript such as the raised two in E=mc².

C - Incorrect: small does not identify the superscript placement required by the example.

D - Incorrect: em marks stress emphasis rather than superscript position.

Primary note proof

Study-note section WEB06

> <sub>/<sup> mark sub/superscripts.

Question sources

Web Programming review questions · Question 19

Question 275 | NS0819

What is the purpose of HTML comments?

A. They are displayed on the webpage

B. They style the content

C. They are ignored by the browser and add notes

D. They link to external files

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C: Comments add source notes without appearing as ordinary page content.

WEB06 states that <!-- comment --> is not displayed but remains visible in source. Thus C's ignored wording should mean ignored for ordinary visual rendering, not removed from the DOM or hidden from source inspection. Comments do not themselves style content or link an external file, so the other purposes do not match.

Why each choice is right or wrong

A - Incorrect: HTML comments are not displayed as ordinary document content.

B - Incorrect: Comments do not supply CSS declarations or style the content by themselves.

C - Correct: Comments hold source notes and are not normally rendered; they remain visible in source.

D - Incorrect: A comment is not the markup mechanism for linking external resources.

Primary note proof

Study-note section WEB06

> <!-- comment --> is not displayed but remains visible in source.

Question sources

Web Programming review questions · Question 20

Question 276 | NS0820

Which JavaScript keyword declares a block-scoped variable?

A. var

B. let

C. def

D. dim

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B: let

WEB15 explicitly distinguishes function/global var from block-scoped let and const. Of the offered names, let declares a block-scoped variable. const would additionally prevent rebinding but is not offered here. def and dim are not JavaScript declaration keywords in the supplied syntax. Therefore B matches the requested scope behavior.

Why each choice is right or wrong

A - Incorrect: var has function/global scope rather than the block scope requested.

B - Correct: let declares a block-scoped JavaScript variable.

C - Incorrect: def is not the JavaScript variable-declaration keyword defined in the note.

D - Incorrect: dim is not the JavaScript variable-declaration keyword defined in the note.

Primary note proof

Study-note section WEB15

> var has function/global scope; let/const are block-scoped.

Question sources

Web Programming review questions · Question 21

Question 277 | NS0821

Which JavaScript keyword declares a constant that cannot be reassigned?

A. static

B. final

C. var

D. const

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer D: const

WEB15 says const prevents rebinding but does not prevent mutation of an object's contents. That is the precise constant-binding property in the stem. var permits reassignment, while static and final are not the declaration form described here. D therefore answers the binding question without claiming that every object assigned to const becomes deeply immutable.

Why each choice is right or wrong

A - Incorrect: static is not the const variable-declaration form for preventing rebinding.

B - Incorrect: final is not the JavaScript binding declaration defined in WEB15.

C - Incorrect: var permits reassignment and does not provide a constant binding.

D - Correct: const prevents rebinding, though object contents can still be mutated.

Primary note proof

Study-note section WEB15

> var has function/global scope; let/const are block-scoped.

Study-note section WEB15

> const prevents rebinding but not mutation of an object's contents.

Question sources

Web Programming review questions · Question 22

Question 278 | NS0822

How are single-line comments written in JavaScript?

A. <!-- comment -->

B. /* comment */

C. // comment

D. # comment

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer C: // comment

WEB15 defines // comments as ending at the line end and /* */ as block comments. Therefore C is the single-line comment syntax. A block comment may be physically written on one line, but it remains a different delimiter form and can span lines. HTML and hash-style markers are not the single-line syntax requested in the note.

Why each choice is right or wrong

A - Incorrect: The shown marker is an HTML comment form rather than the standard JavaScript single-line form.

B - Incorrect: This is block-comment syntax, even when physically written on one line.

C - Correct: Double slash starts a JavaScript comment that ends at the line end.

D - Incorrect: A hash is not the ordinary JavaScript single-line comment delimiter given in WEB15.

Primary note proof

Study-note section WEB15

> // comments end at line end; /* */ span a block.

Study-note section WEB06

> <!-- comment --> is not displayed but remains visible in source.

Question sources

Web Programming review questions · Question 23

Question 279 | NS0823

A form uses onsubmit="return validate()". If validate() returns false, what happens to the cancelable default submission?

A. Submits the form

B. Refreshes the page

C. Prevents form submission

D. Clears all form fields

Hint

Separate event dispatch, handler registration and cancellation of the browser default action.

Answer C: Prevents form submission

The inline onsubmit convention consumes the value returned by validate. Returning false therefore cancels the form's default submission in this stated setup. It does not clear fields or explicitly refresh the page. The caller matters: returning false from a callback installed through addEventListener would not by itself cancel submission and would require preventDefault instead. The revised question includes the handler convention needed for the definite result.

Why each choice is right or wrong

A - Incorrect: The consumed false result cancels rather than initiates default submission.

B - Incorrect: No page-refresh action is requested by returning false.

C - Correct: The inline onsubmit return convention cancels the default submission.

D - Incorrect: Returning a Boolean does not clear control values.

Primary note proof

Study-note section WEB21

> Returning false from an addEventListener callback does not cancel it; return false works in an appropriate inline/onsubmit property handler whose caller uses that return.

Question sources

Web Programming review questions · Question 24

Question 280 | NS0824

Which event fires when a page finishes loading in JavaScript?

A. onclick

B. onsubmit

C. oninput

D. window.onload

Hint

Separate event dispatch, handler registration and cancellation of the browser default action.

Answer D: window.onload is the handler property for the page load event.

WEB22 says window.onload handles load, typically after dependent resources finish. D therefore names the relevant handler property, although the actual event name is load rather than window.onload. Click, submit, and input refer to other interactions. This distinction avoids confusing a property used to register a handler with the event object or event name itself.

Why each choice is right or wrong

A - Incorrect: onclick handles clicks, not completion of page resource loading.

B - Incorrect: onsubmit handles form submission, not the page load completion event.

C - Incorrect: oninput handles control input changes, not the page load completion event.

D - Correct: window.onload registers the handler for load, normally after dependent resources finish.

Primary note proof

Study-note section WEB22

> window.onload is a handler property for the load event, typically after dependent resources finish.

Study-note section WEB21

> An event handler runs when an event occurs: click, input, keydown, mouseover, submit and load are examples.

Question sources

Web Programming review questions · Question 25

Question 281 | NS0825

In PHP, how do you declare a variable named name with value 'John'?

A. var name = 'John';

B. let $name = 'John';

C. $name = 'John';

D. name = 'John';

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer C: $name = 'John';

WEB23 requires a dollar-prefixed PHP variable name and permits dynamic typing without a preceding type declaration. C correctly assigns the single-quoted string to $name. var name and let mix other declaration syntax with PHP, while bare name lacks the required sigil. No separate JavaScript-style declaration keyword is needed for this ordinary PHP assignment.

Why each choice is right or wrong

A - Incorrect: This is not the ordinary PHP assignment syntax and lacks the dollar-prefixed variable name.

B - Incorrect: let is not the PHP declaration syntax for this assignment.

C - Correct: The dollar-prefixed variable and quoted string assignment are valid PHP syntax.

D - Incorrect: Bare name lacks the required dollar sigil for a PHP variable.

Primary note proof

Study-note section WEB23

> Variables begin with $, then a letter/underscore and subsequent letters/digits/underscores; $AGE18 works but $18AGE does not.

Study-note section WEB23

> Dynamic typing permits values of different types without a preceding type declaration, though optional declarations can constrain them.

Study-note section WEB-S06

> A complete ordinary PHP assignment statement ends with a semicolon.

Question sources

Web Programming review questions · Question 26

Question 282 | NS0826

Which PHP function returns the length of a string?

A. str_word_count()

B. strrev()

C. strlen()

D. strpos()

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: strlen(), measuring bytes.

WEB24 says strlen counts bytes, so it is the matching length function. For ordinary ASCII strings byte count also equals character count; multibyte Unicode can differ. strrev reverses, strpos locates a substring, and a word-count operation does not measure the string's byte length. The qualification prevents treating strlen as a universal user-perceived-character counter.

Why each choice is right or wrong

A - Incorrect: Word counting does not measure the string's byte length.

B - Incorrect: strrev reverses the byte sequence rather than returning its length.

C - Correct: strlen returns byte length, matching the requested function with the Unicode caveat.

D - Incorrect: strpos returns a substring position or false rather than a length.

Primary note proof

Study-note section WEB24

> strlen counts bytes; strpos can return integer0 or false, requiring ===false for failure.

Study-note section WEB24

> strrev reverses bytes and str_replace replaces matching content.

Question sources

Web Programming review questions · Question 27

Question 283 | NS0827

What does strrev("Hello") return in PHP?

A. Hello

B. hello

C. HELLO

D. olleH

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer D: olleH

WEB24 defines strrev as reversing bytes. Hello is ASCII, so reversing its five bytes gives o,l,l,e,H in that order, retaining the original uppercase H at the final position. The function does not perform lowercase or uppercase conversion and does not leave the sequence unchanged. D is the exact returned string.

Why each choice is right or wrong

A - Incorrect: Hello is unchanged, whereas strrev reverses the byte order.

B - Incorrect: Lowercasing is not the operation performed by strrev.

C - Incorrect: Uppercasing is not the operation performed by strrev.

D - Correct: Reversing the ASCII bytes H,e,l,l,o yields o,l,l,e,H.

Primary note proof

Study-note section WEB24

> strrev reverses bytes and str_replace replaces matching content.

Question sources

Web Programming review questions · Question 28

Question 284 | NS0828

Which PHP function finds the position of a substring within a string?

A. str_replace()

B. strpos()

C. strlen()

D. strrev()

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: strpos()

WEB24 identifies strpos as the substring-position function and warns that a successful position zero differs from false. Use strict ===false when checking failure so an initial-position match is not mistaken for absence. Replacement, length, and reversal functions perform different operations. Thus B names the lookup API rather than a transformation function.

Why each choice is right or wrong

A - Incorrect: str_replace changes matching content rather than reporting the match position.

B - Correct: strpos finds a substring position, returning zero legitimately or false for failure.

C - Incorrect: strlen measures byte length rather than locating a substring.

D - Incorrect: strrev reverses bytes instead of reporting a match location.

Primary note proof

Study-note section WEB24

> strlen counts bytes; strpos can return integer0 or false, requiring ===false for failure.

Study-note section WEB24

> strrev reverses bytes and str_replace replaces matching content.

Question sources

Web Programming review questions · Question 29

Question 285 | NS0829

What is the key difference between include and require in PHP?

A. include is faster than require

B. require gives a fatal error if file is missing; include gives a warning

C. include gives a fatal error; require gives a warning

D. There is no difference

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: include warns; failed require raises Error in PHP8 and stops normal execution if uncaught.

WEB33 distinguishes include's warning from require's critical failure. B captures the conventional contrast when require's failure is uncaught, but PHP8 raises a catchable Error rather than the older E_COMPILE_ERROR description. The distinction is failure behavior, not an inherent speed ranking. C reverses the behaviors and D denies the explicit difference.

Why each choice is right or wrong

A - Incorrect: No inherent speed advantage is the failure-handling distinction defined by WEB33.

B - Correct: This is the conventional contrast for an uncaught failure; PHP8 require raises a catchable Error.

C - Incorrect: This reverses include's warning and require's critical failure behavior.

D - Incorrect: The note explicitly distinguishes their behavior when the target file cannot be included.

Primary note proof

Study-note section WEB33

> include evaluates another file in the current scope and warns on failure; require fails critically.

Study-note section WEB33

> In PHP8 a failed require raises Error, which can be caught; uncaught it stops normal execution.

Question sources

Web Programming review questions · Question 30

Question 286 | NS0830

Which PHP superglobal is dedicated specifically to supported form data submitted through POST, rather than a configurable mixture of request sources?

A. $_GET

B. $_SESSION

C. $_POST

D. $_REQUEST

Hint

Identify the data source or security responsibility precisely, and check whether a combined source or a different control could also qualify.

Answer C: $_POST

$_POST directly represents supported form POST values. $_REQUEST can include POST values under some configurations, but it combines sources and is not dedicated to the form POST channel. $_GET represents URL query values and $_SESSION contains application session state. The source-specific wording makes C unique while preserving the fact that a configurable combined array may also expose some of the same submitted values.

Why each choice is right or wrong

A - Incorrect: $_GET supplies URL query values.

B - Incorrect: $_SESSION supplies stored session state rather than the current form POST source.

C - Correct: $_POST is dedicated to supported form POST data.

D - Incorrect: $_REQUEST may combine POST with other configured sources.

Primary note proof

Study-note section WEB27

> $_GET contains query parameters; $_POST contains supported form POST data; $_FILES upload metadata; $_SERVER request/environment details; $_COOKIE cookie values; $_SESSION session data after startup.

Study-note section WEB27

> $_REQUEST may combine GET/POST/cookie data according to configuration; it is not a database interface and has ambiguous precedence, so prefer the intended source.

Question sources

Web Programming review questions · Question 31

Question 287 | NS0831

Which SQL statement retrieves data from a database?

A. INSERT INTO

B. UPDATE

C. DELETE

D. SELECT

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer D: SELECT

WEB32 directly assigns retrieval to SELECT, adding to INSERT, changing values to UPDATE, and removal to DELETE. Therefore D is the SQL statement role requested. This identifies the operation's semantics; the actual query still needs appropriate columns, filters, authorization, and safe parameter handling when incorporating submitted values.

Why each choice is right or wrong

A - Incorrect: INSERT INTO adds rows rather than naming the ordinary retrieval statement.

B - Incorrect: UPDATE changes stored values rather than the retrieval operation requested.

C - Incorrect: DELETE removes rows rather than retrieving them.

D - Correct: SELECT is explicitly the SQL operation that retrieves rows.

Primary note proof

Study-note section WEB32

> SELECT retrieves rows; INSERT adds; UPDATE ...

Question sources

Web Programming review questions · Question 32

Question 288 | NS0832

What keyword assigns new values in a SQL UPDATE statement?

A. VALUES

B. WHERE

C. SET

D. INTO

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer C: SET

WEB32 gives UPDATE ... SET as the form that changes values. SET introduces the target-column assignments, while WHERE limits which rows are affected. VALUES and INTO occur in other statement roles and do not replace SET's assignment function. C therefore identifies the keyword requested, distinct from selecting the affected rows.

Why each choice is right or wrong

A - Incorrect: VALUES does not introduce assignments in the UPDATE ... SET form given here.

B - Incorrect: WHERE selects matching rows rather than assigning the new column values.

C - Correct: SET introduces the value assignments in an UPDATE statement.

D - Incorrect: INTO does not replace SET for assignments in the stated UPDATE form.

Primary note proof

Study-note section WEB32

> SELECT retrieves rows; INSERT adds; UPDATE ...

Study-note section WEB32

> WHERE limits affected/matching records; omitting it from DELETE ordinarily targets every row, subject to constraints, permissions and errors.

Question sources

Web Programming review questions · Question 33

Question 289 | NS0833

What happens if you use DELETE without a WHERE clause?

A. Nothing happens

B. Only the first record is deleted

C. An error is generated

D. All records in the table are deleted

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer D: It ordinarily targets all rows for deletion, subject to constraints, permissions, and execution success.

WEB32 says WHERE limits affected rows and omitting it from DELETE ordinarily targets every row. Thus D is the intended effect of a successful unrestricted DELETE. No first-row-only rule exists, and omission is not inherently a syntax error. Permissions, constraints, or other errors can still prevent the operation, so the outcome is not an unconditional execution guarantee.

Why each choice is right or wrong

A - Incorrect: Without a filter, the statement ordinarily targets rows rather than doing nothing.

B - Incorrect: There is no implicit first-row-only restriction when WHERE is omitted.

C - Incorrect: Omitting WHERE is not by itself the stated syntax error.

D - Correct: A successful unrestricted DELETE ordinarily deletes all targeted table rows, subject to database controls.

Primary note proof

Study-note section WEB32

> WHERE limits affected/matching records; omitting it from DELETE ordinarily targets every row, subject to constraints, permissions and errors.

Question sources

Web Programming review questions · Question 34

Question 290 | NS0834

Which PHP function establishes a connection to MySQL?

A. mysql_open()

B. db_connect()

C. mysqli_connect()

D. connect_mysql()

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer C: mysqli_connect()

WEB31 explicitly gives mysqli_connect(host,user,password,database,port,socket) as the MySQLi connection function. The other offered names are not the API identified here. Argument defaults and actual credentials affect connection success, so knowing the function name does not remove the need for configuration and failure handling. C is the valid connection entry point.

Why each choice is right or wrong

A - Incorrect: mysql_open() is not the MySQLi connection function given in WEB31.

B - Incorrect: db_connect() is not the standard MySQLi connection function identified here.

C - Correct: mysqli_connect() is explicitly the MySQLi connection-opening function in the note.

D - Incorrect: connect_mysql() is not the offered standard API name described by WEB31.

Primary note proof

Study-note section WEB31

> mysqli_connect(host,user,password,database,port,socket) opens a connection; selecting a database can be deferred, and defaults make several arguments technically optional.

Question sources

Web Programming review questions · Question 35

Question 291 | NS0835

Which PHP function returns the ID of the last inserted record?

A. mysqli_last_id()

B. mysqli_insert_id()

C. mysqli_get_id()

D. mysqli_row_id()

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer B: mysqli_insert_id($conn), for a generated AUTO_INCREMENT value on that connection.

WEB32 identifies mysqli_insert_id($conn) as reporting the connection's generated AUTO_INCREMENT value. It is not a universal lookup of an arbitrary last row or a promise of gapless IDs. B names the correct function; the others invent different names. A connection argument and appropriate successful insertion context are needed in actual procedural code.

Why each choice is right or wrong

A - Incorrect: mysqli_last_id() is not the generated-value function named in WEB32.

B - Correct: mysqli_insert_id($conn) reports the connection's generated AUTO_INCREMENT value.

C - Incorrect: mysqli_get_id() is not the API identified for the generated insert value.

D - Incorrect: mysqli_row_id() is not the API named for a generated insert identifier.

Primary note proof

Study-note section WEB32

> mysqli_num_rows counts a result's rows; mysqli_affected_rows($conn) counts changed/inserted/deleted rows under its options; mysqli_insert_id($conn) reports a generated AUTO_INCREMENT value from the connection.

Study-note section WEB32

> AUTO_INCREMENT generates numeric values but gaps and explicit values are possible; do not promise gapless consecutive business numbering.

Question sources

Web Programming review questions · Question 36

Question 292 | NS0836

Which ordinary PHP variable scope can be accessed inside a function using a global declaration or, alternatively, the $GLOBALS array?

A. Local

B. Static

C. Global

D. Session

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer C: Global

A variable declared outside functions has global scope. A function normally has separate local bindings, so it can explicitly import such a binding with global or access it through $GLOBALS. The keyword is therefore one mechanism, not a mandatory exclusive spelling. A static local remains a function-local binding whose value persists across calls; session state belongs to a superglobal available across scopes.

Why each choice is right or wrong

A - Incorrect: Ordinary local bindings already belong to that function; this is not importing an outside binding.

B - Incorrect: A static local retains its value across calls but remains local to the function.

C - Correct: Global bindings are the outside-function bindings accessed by these two mechanisms.

D - Incorrect: Session state uses a superglobal available across scopes, rather than this ordinary binding import.

Primary note proof

Study-note section WEB25

> A variable outside a function has global scope.

Study-note section WEB25

> A function’s locals and by-value parameters are separate bindings; use global $x or $GLOBALS["x"] to refer to a global explicitly.

Study-note section WEB25

> A static local retains its value across calls during the execution lifecycle; static storage is not simply another visibility scope.

Study-note section WEB25

> Superglobals are available across scopes.

Question sources

Web Programming review questions · Question 37

Question 293 | NS0837

What is a PHP static variable?

A. A variable that cannot be changed

B. A global variable accessible everywhere

C. A local variable that retains its value between function calls

D. A variable stored in a session

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer C: A local variable that retains its value between calls during the execution lifecycle.

WEB25 defines a static local by persistence across function calls. It remains a local binding and can change value; static is not the same as constant, global visibility, or server session storage. Thus C supplies both the local-scope and retained-state aspects, without treating static duration as an independent universal visibility scope.

Why each choice is right or wrong

A - Incorrect: Static locals can change value; static does not mean immutable.

B - Incorrect: Static persistence does not make the variable a global visible everywhere.

C - Correct: A static local retains its value across calls during the execution lifecycle.

D - Incorrect: A static local is not the same as data stored in $_SESSION.

Primary note proof

Study-note section WEB25

> A static local retains its value across calls during the execution lifecycle; static storage is not simply another visibility scope.

Study-note section WEB23

> Constants cannot normally be redefined during execution.

Question sources

Web Programming review questions · Question 38

Question 294 | NS0838

What does var_dump() do in PHP?

A. Deletes a variable

B. Displays data type and value of a variable

C. Converts a variable to a string

D. Validates a variable

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer B: Display the variable's type and value.

WEB23 explicitly states that var_dump displays type and value. It is diagnostic output, not a deletion, validation, or conversion guarantee. Observing a value can assist debugging, but it does not prove that the input satisfies application constraints. B is therefore the direct API purpose, while the alternatives describe unrelated operations.

Why each choice is right or wrong

A - Incorrect: var_dump does not delete the variable whose diagnostic information it displays.

B - Correct: The note explicitly defines var_dump as displaying type and value.

C - Incorrect: Diagnostic display is not a conversion of the original variable binding to a string.

D - Incorrect: Displaying a value does not establish application-specific validation.

Primary note proof

Study-note section WEB23

> var_dump displays type and value.

Question sources

Web Programming review questions · Question 39

Question 295 | NS0839

Which PHP data type represents the absence of a value?

A. Boolean

B. String

C. Integer

D. NULL

Hint

Check exact language syntax, declaration rules and execution context before reasoning about the result.

Answer D: NULL

WEB23 says null means no value. Boolean represents true or false, String text, and integer numeric values; none is the same absence marker. In particular zero, false, and an empty string can be legitimate present values rather than null. D therefore matches the requested absence-of-value representation.

Why each choice is right or wrong

A - Incorrect: A Boolean is a true/false value rather than the absence marker.

B - Incorrect: A string, including an empty one, is not the null absence marker.

C - Incorrect: An integer such as zero is still a value rather than null.

D - Correct: NULL is explicitly the no-value representation identified in WEB23.

Primary note proof

Study-note section WEB23

> Scalars include int,float,string,bool; arrays/objects are structured values and null means no value.

Question sources

Web Programming review questions · Question 40

Question 296 | NS0840

For ordinary HTML form submission, which statement correctly contrasts where POST places form data with GET's URL query?

A. POST is faster than GET

B. POST data is visible in the URL

C. POST form data normally goes in the request body rather than the URL query, subject to configured size limits

D. POST is easier to implement

Hint

Identify the protocol layer and the intended request, response or channel semantics.

Answer C: POST form data normally goes in the request body rather than the URL query, subject to configured size limits

Ordinary GET forms place name/value pairs in the URL query, whereas POST forms place them in the request body. This is a placement distinction, not encryption: tools can inspect the body, and servers impose configured limits. The other claims either put POST fields in the wrong location or assert universal speed/ease advantages that do not follow from the method semantics.

Why each choice is right or wrong

A - Incorrect: Method choice does not guarantee that a POST operation is faster; this is not the placement contrast asked for.

B - Incorrect: Ordinary POST form values are placed in the body, not the URL query used by GET forms.

C - Correct: This gives the body-versus-query distinction while retaining configured limits and making no secrecy claim.

D - Incorrect: Ease of implementation is application-dependent and does not describe the form-data placement difference.

Primary note proof

Study-note section WEB04

> GET forms usually put name/value pairs in the URL query; POST forms put them in the body.

Study-note section WEB04

> Neither method encrypts data.

Study-note section WEB04

> POST bodies are visible to developer tools and subject to server/configured size limits.

Question sources

Web Programming review questions · Question 41

Question 297 | NS0841

Which HTML form input type masks characters for security?

A. type="text"

B. type="hidden"

C. type="secure"

D. type="password"

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer D: type="password"

WEB09 says password inputs mask display but do not encrypt the transmitted value. D therefore supplies the intended visual masking. Hidden inputs remove visible interaction rather than offering a masked text-entry control, text displays characters normally, and secure is not the type defined here. Transport and server protection remain separate obligations.

Why each choice is right or wrong

A - Incorrect: A text input displays entered characters normally rather than applying password masking.

B - Incorrect: A hidden input is not an interactive masked-character text field.

C - Incorrect: secure is not the password input type defined in the note.

D - Correct: password masks displayed characters while leaving encryption to separate transport controls.

Primary note proof

Study-note section WEB09

> type="password" masks display but does not encrypt.

Study-note section WEB-S02

> A hidden input carries a value without displaying an editable text field; it is not a password-masking control.

Question sources

Web Programming review questions · Question 42

Question 298 | NS0842

What attribute groups radio buttons so only one can be selected?

A. id

B. value

C. name

D. group

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer C: name

WEB09 explicitly groups same-name radio buttons for a single choice. id instead identifies an individual element and connects a label; value is the submitted selected value. group is not the grouping attribute defined here. Therefore C gives the shared attribute that creates the radio-choice group under the form-control model.

Why each choice is right or wrong

A - Incorrect: id identifies an individual control and associates labels rather than defining the shared radio group.

B - Incorrect: value supplies the selected control’s submitted value rather than the grouping key.

C - Correct: Radio buttons sharing the same name form a single-choice group.

D - Incorrect: group is not the radio-grouping attribute defined in WEB09.

Primary note proof

Study-note section WEB09

> A label’s for matches the control’s id; name identifies its submitted field.

Study-note section WEB09

> Same-name radio buttons form a single-choice group; checkboxes permit independent selections.

Study-note section WEB-S02

> For a text input, the value attribute supplies its initial text value; it does not specify the control’s width.

Question sources

Web Programming review questions · Question 43

Question 299 | NS0843

Which HTML element creates a dropdown menu?

A. <input type="dropdown">

B. <dropdown>

C. <select>

D. <menu>

Hint

Separate the control role, displayed content, submitted value and form destination.

Answer C: <select> containing <option> elements.

WEB09 defines select with option children as the choice control, conventionally rendered as a dropdown in its ordinary single-selection form. The input type dropdown and dropdown element are not the provided control syntax. menu is not the form-selection element requested. Additional attributes can alter select presentation, but the native form-control answer remains C.

Why each choice is right or wrong

A - Incorrect: dropdown is not the input type provided for this native selection control.

B - Incorrect: dropdown is not the native select element spelling defined by WEB09.

C - Correct: select with option children creates the native choice control commonly displayed as a dropdown.

D - Incorrect: menu is not the form-selection control defined by WEB09.

Primary note proof

Study-note section WEB09

> <select> contains <option>; textarea accepts multiple lines.

Question sources

Web Programming review questions · Question 44

Question 300 | NS0844

What does the accept attribute on a file-upload input do?

A. Automatically uploads the file

B. Hints which file types the browser’s file picker should offer

C. Sets the maximum file size

D. Validates the file name

Hint

Identify the role requested and compare it with the definitions and constraints in the cited passages.

Answer B: Hints which file types the browser’s file picker should offer

accept is a browser selection hint describing file types the picker should offer. It does not guarantee what an attacker can submit or what the server actually accepts. Server-side validation must separately check upload errors, size and content before safe storage. The attribute also neither initiates transfer, sets a maximum upload size nor validates a filename. The choice states the UI role without claiming server enforcement.

Why each choice is right or wrong

A - Incorrect: The attribute does not automatically start an upload.

B - Correct: This is the browser-picker hint described in the note.

C - Incorrect: Maximum upload size requires separate controls and validation.

D - Incorrect: The attribute does not validate filenames.

Primary note proof

Study-note section WEB30

> accept hints to the browser which file types to offer; it does not enforce the server’s policy.

Study-note section WEB30

> Validate server-side size/content, inspect upload errors, use the temporary upload path, choose a safe generated destination and use move_uploaded_file.

Question sources

Web Programming review questions · Question 45

Question 301 | NS0845

Which listed HTML elements can embed supported video or other multimedia resources in a suitable environment? Select all that apply.

A. <video>

B. <media>

C. <embed>

D. <object>

Hint

Use the element or attribute meaning and document structure, rather than appearance or a similar-looking name.

Answer A, C, D: <video>; <embed>; <object>

video is the native HTML element for video, and embed and object can also embed supported resources, including multimedia in a suitable environment. The question allows these different embedding mechanisms rather than asking only for the native video element. media is not the corresponding HTML element. Browser support, resource formats and appropriate permissions still matter; the selection describes capability rather than guaranteed playback of every resource.

Why each choice is right or wrong

A - Correct: video is the native HTML video element.

B - Incorrect: media is not the HTML multimedia-embedding element identified here.

C - Correct: embed can embed supported multimedia resources.

D - Correct: object can embed supported multimedia resources.

Primary note proof

Study-note section WEB10

> Modern audio/video elements embed supported media, often with controls and fallback content; embed/object can also embed supported resources.

Question sources

Web Programming review questions · Question 46

Question 302 | NS0846

What does session_start() do in PHP?

A. Creates a new database connection

B. Starts or resumes a session to store user data

C. Starts an HTML form

D. Initializes a new variable

Hint

Follow the identifier and server-side state through the stated lifecycle or access check; distinguish client-controlled claims from server decisions.

Answer B: Start or resume PHP session handling and make $_SESSION available.

WEB29 directly defines session_start() as starting or resuming session handling. The browser normally holds an opaque identifier while server-controlled storage holds the session data. This is separate from a database connection or an HTML form. It does not simply initialize one arbitrary variable, and the server still must validate authorization for protected actions.

Why each choice is right or wrong

A - Incorrect: Session startup is separate from opening a database connection.

B - Correct: session_start starts or resumes PHP session handling and makes $_SESSION available.

C - Incorrect: It is not an HTML form-starting operation.

D - Incorrect: It initializes session handling rather than merely declaring one arbitrary variable.

Primary note proof

Study-note section WEB29

> session_start() starts/resumes PHP session handling and makes $_SESSION available.

Study-note section WEB29

> Normally the browser holds an opaque session identifier while data resides in server-controlled storage.

Question sources

Web Programming review questions · Question 47

Question 303 | NS0847

In CSS, which selector type is denoted by a dot (.)?

A. ID selector

B. Element selector

C. Group selector

D. Class selector

Hint

Identify which elements a rule targets and distinguish selector syntax from declaration syntax.

Answer D: Class selector

WEB11 defines .notice as a class selector and #logo as an ID selector. The dot is therefore the prefix for a class name, while an element selector uses its type name and grouping uses commas. D correctly identifies the selector category rather than a declaration property or an element's unique ID.

Why each choice is right or wrong

A - Incorrect: An ID selector uses # rather than a dot.

B - Incorrect: An element/type selector uses the element name without the class prefix.

C - Incorrect: Grouping uses commas between selectors, not a dot.

D - Correct: A dot prefixes a class selector, as in .notice.

Primary note proof

Study-note section WEB11

> Type p selects paragraph elements, .notice selects a class, #logo an ID, h1,h2 groups selectors, nav ul selects descendants and nav>ul direct children.

Question sources

Web Programming review questions · Question 48

Question 304 | NS0848

What does the SQL AUTO_INCREMENT property do?

A. Auto-updates a column when a record is modified

B. Generates a unique sequential number for a column

C. Automatically deletes old records

D. Automatically backs up the database

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer B: Generate automatic numeric values, commonly used for keyed row identifiers; gaps are possible.

WEB32 says AUTO_INCREMENT generates numeric values and permits gaps or explicit values. B is the intended generated-number description when used with the normal keyed column design, but sequential must not mean gapless business numbering or a promise about every manually supplied value. It does not update arbitrary columns, delete records, or back up the database.

Why each choice is right or wrong

A - Incorrect: AUTO_INCREMENT does not automatically update an arbitrary column whenever a record is modified.

B - Correct: It generates numeric values commonly used for row identifiers; gaps and explicit values remain possible.

C - Incorrect: It does not automatically delete old rows.

D - Incorrect: It does not perform database backups.

Primary note proof

Study-note section WEB32

> AUTO_INCREMENT generates numeric values but gaps and explicit values are possible; do not promise gapless consecutive business numbering.

Question sources

Web Programming review questions · Question 49

Question 305 | NS0849

Which PHP function closes a MySQL database connection?

A. mysqli_end()

B. mysqli_disconnect()

C. mysqli_close()

D. db_close()

Hint

Identify the API and operation first, then distinguish connection setup, statement execution and result processing.

Answer C: mysqli_close($conn)

WEB32 explicitly gives mysqli_close as closing a connection. C names that MySQLi API; the procedural form uses the connection argument in real code. Closing a connection is separate from ending a query or deleting stored data, and the other offered function names are not the standard one identified here.

Why each choice is right or wrong

A - Incorrect: mysqli_end() is not the connection-closing function named in WEB32.

B - Incorrect: mysqli_disconnect() is not the API identified here for this operation.

C - Correct: mysqli_close closes the MySQLi connection, using its connection argument in procedural code.

D - Incorrect: db_close() is not the standard MySQLi function named in the supplied note.

Primary note proof

Study-note section WEB32

> Escape rendered cell text; use prepared statements for submitted values; mysqli_close closes the connection.

Question sources

Web Programming review questions · Question 50

Operating System

Questions & Worked Answers

233 questions + 1 source variants

Question 1 | Q00023

Which mutual exclusion mechanism keeps track of turns to enter the critical region using an integer variable, which is initially-0, and another process waits until it turns to 1?

A. Lock variable

B. Disabling interrupts

C. Strict alteration

D. Race condition

Hint

Track who may enter and distinguish separate tests and updates from atomic operations.

Answer C: Strict alternation (option misspells it as alteration).

OS12 defines strict alternation using turn=0/1: each process waits until turn equals its identifier and passes the turn on leaving. That directly matches the described integer ownership variable. A naive lock variable instead records availability and can race during separate test/set operations. Disabling interrupts and a race condition do not implement the described turn-taking algorithm.

Why each choice is right or wrong

A - Incorrect: A naive lock variable records free/busy status rather than assigning alternating ownership to process 0 or 1.

B - Incorrect: Disabling interrupts does not implement the integer turn-owner protocol described in the stem.

C - Correct: Strict alternation uses a turn variable initialized to one process and passed to the other.

D - Incorrect: A race condition is a concurrency failure, not a mechanism providing the described alternating access.

Primary note proof

Study-note section OS12

> Strict alternation uses turn=0/1: process i waits while turn≠i, then enters and sets turn to the other process on leaving. It enforces turns but can violate progress if the other process stays outside its critical region. A naive lock-variable test followed by a separate set races because both processes can observe free before either sets it.

Study-note section OS12

Test-and-set and compare-and-swap combine checking/updating atomically; ordinary check-then-write does not. Peterson’s classic algorithm uses flags plus turn for two processes under its memory-model assumptions; a semaphore can coordinate more than two. Disabling local interrupts does not exclude other CPUs and is not a general user-space locking solution.

Study-note section OS10

A race occurs when results depend on uncoordinated interleaving of conflicting accesses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

• Loads a program into memory.
• Executes the program.
• Handles program's execution.
• Provides a mechanism for process synchronization.
• Provides a mechanism for process communication.
• Provides a mechanism for deadlock handling.

Operating System module | PDF page 45 | lines 10-22

The CPU hardware uses an interrupt request line wire which helps CPU to sense after executing every instruction. When the CPU checks that a controller has put a signal on the interrupt request line, the CPU saves a state, such as the current value of the instruction pointer, and jumps to the interrupt handler routine at a fixed address. The interrupt handler part determines the cause of the interrupt performs the necessary processing and executes an interrupt instruction to return the CPU to its execution state.
The basic mechanism of interrupt enables the CPU to respond to an asynchronous event, such as when a device controller becomes ready for service. Most CPUs have two interrupt request lines.
• non-maskable interrupt - Such kind of interrupts are reserved for events like unrecoverable memory errors.
• maskable interrupt - Such interrupts can be switched off by the CPU before the execution of critical instructions that must not be interrupted.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 23

Question 2 | Q00031

Which one of the following messaging communication best explains the statement: "Until the message is received, the sending process is blocked'"?

A. Blocking receive

B. Non-block receive

C. Non-block sending

D. Blocking send

Hint

Identify how data moves, which participant waits and what event allows the operation to continue.

Answer D: Blocking send.

OS09 explicitly identifies a sender blocked until receipt as blocking send under rendezvous semantics. The stem describes the sending process waiting, not a receiver awaiting an incoming message. Nonblocking sending returns promptly rather than waiting for receipt, and receive modes describe the other side. The receipt condition here supplies the relevant API assumption.

Why each choice is right or wrong

A - Incorrect: Blocking receive waits on the receiving side, whereas the stem explicitly blocks the sender.

B - Incorrect: Nonblocking receive returns promptly on the receiving side and does not describe sender waiting.

C - Incorrect: Nonblocking sending returns promptly rather than keeping the sender blocked until the handoff completes.

D - Correct: Under the stated rendezvous condition, waiting until message receipt is blocking-send behavior.

Primary note proof

Study-note section OS09

> Blocking send waits for the required handoff/buffering/receipt condition defined by its API. Blocking receive waits for a message. Nonblocking operations return promptly with data, progress or a would-block result. “Sender blocked until received” identifies blocking send under rendezvous semantics.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 11 | lines 15-19

> • Two processes often require data to be transferred between them.
> • The both processes can be on the one computer or on different computer but are connected through computer network.
> • Communication may be implemented by two methods either by Shared Memory or by Message Passing.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 31

Question 3 | Q00034

_____is the component of the operating system that is responsible to decide which process to run first among ready processes.

A. Memory manager

B. Inter process communication

C. Disk manager

D. Scheduler

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer D: Scheduler

The CPU scheduler selects among ready processes and allocates the CPU (PDF p20). This is precisely the selection task in the stem.

Why each choice is right or wrong

A - Incorrect: Memory allocation is a separate function (p5).

B - Incorrect: Process data exchange (p11) is not selection of the next CPU user.

C - Incorrect: Device allocation (p6) does not decide CPU order.

D - Correct: CPU scheduling selects ready processes (p20).

Primary note proof

Study-note section OS06

> The ready queue contains eligible work; device queues hold blocked requests. The broad textbook job queue contains processes in the system, while an admission queue more narrowly holds jobs awaiting memory/admission. Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work. The dispatcher performs the handoff/context restore after selection. A bare “priority” causes running→ready only if a preemptive policy chooses another process.

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs.

Study-note section OS09

> Shared memory lets processes access a common region and requires synchronization for conflicting updates. Message passing exchanges data through send/receive operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 20 | lines 15-21

It is also called CPU scheduler. Main objective is increasing system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects process among the processes that are ready to execute and allocates CPU to one of them.
Short term scheduler also known as dispatcher, execute most frequently and makes the fine-grained decision of which process to execute next. Short term scheduler is faster than long term scheduler.

Operating System module | PDF page 5 | lines 26-36

Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address. Main memory provides a fast storage that can be access directly by the CPU. So, for a program to be executed, must in the main memory.
Activities Operating System for memory management.
Keeps tracks of primary memory (i.e. what part of it are in use by whom, what part are not in use.)
In multiprogramming, OS decides which process will get memory when and how much.
Allocates the memory when the process requests it to do so.
De-allocates the memory when the process no longer needs it or has been terminated.

Operating System module | PDF page 11 | lines 15-19

• Two processes often require data to be transferred between them.
• The both processes can be on the one computer or on different computer but are connected through computer network.
• Communication may be implemented by two methods either by Shared Memory or by Message Passing.

Operating System module | PDF page 6 | lines 11-17

OS manages device communication via their respective drivers. Activities of Operating System for device management.
Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.
Decides which process gets the device when and for how much time.
Allocates the device in the efficient way.
De-allocates devices.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 34

Question 4 | Q00092

Which one of the following is not true?

A. kernel is the program that constitutes the central core of the operating system

B. kernel is the first part of operating system to load into memory during booting

C. kernel is made of various modules which cannot be loaded in running operating system

D. kernel remains in the memory during the entire computer session

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer C: The claim that kernel modules cannot load while running is false.

OS03 states that loadable modules can add drivers to a running kernel, directly contradicting option C's absolute prohibition. It describes the kernel as the core OS component and resident through the session. Option B is read as the first core OS component, rather than the first code executed at power-on: firmware and bootloader stages precede it.

Why each choice is right or wrong

A - Incorrect: The kernel implements core resource management, protection and system-call handling, so this statement is true.

B - Incorrect: As the first core OS component, the kernel is loaded at boot; firmware/bootloader execution precedes that OS boundary.

C - Correct: Modular kernels permit runtime module loading, directly refuting the statement that it cannot occur.

D - Incorrect: A resident kernel remains active throughout the session, matching the note's ordinary kernel model.

Primary note proof

Study-note section OS03

> The kernel implements core resource management, protection and system-call handling.

Study-note section OS03

> Firmware/bootloader stages execute before transferring control to a kernel loaded into RAM. The kernel is the first core OS component, not the first program ever executed at power-on. Loadable modules can add drivers to a running kernel; "kernel modules cannot be loaded while running" is false in modular systems. A resident kernel remains active through the session. Utilities such as antivirus and disk tools can combine user processes with privileged components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 11-15

operating system. It is consisting of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
System Library - System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries

Operating System module | PDF page 32 | lines 11-14

In dynamic loading, a routine of a program is not loaded until it is called by the program. All routines are kept on disk in a re-locatable load format. The main program is loaded into memory and is executed. Other routines methods or modules are loaded on request. Dynamic loading makes better memory space utilization and unused routines are never loaded.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 92

Computer Science model exam · Question 95

University of Gondar model exam · Question 95

Question 5 | Q00164

One is not the goal of I/O software

A. Buffering

B. Device independence

C. Uniform naming

D. Error handling

E. None of the above

Hint

Separate device hardware, device-specific software and general event or transfer mechanisms.

Answer E: None: all four are I/O-software goals or supporting functions.

OS34 identifies device independence, uniform naming and error handling as I/O services. OS35 explains buffering as holding in-transit data to absorb speed/size differences. Each listed feature therefore supports I/O software, so none is an exclusion. Calling buffering a supporting mechanism rather than a high-level objective does not make it unrelated to I/O software’s goals.

Why each choice is right or wrong

A - Incorrect: Buffering absorbs transfer-speed and size differences, directly supporting reliable and efficient I/O software.

B - Incorrect: Device independence is an explicit I/O service that hides device-specific details from higher layers.

C - Incorrect: Uniform naming is explicitly included among the I/O services described here.

D - Incorrect: Reporting or retrying supported device failures is the error-handling responsibility of I/O software.

E - Correct: All four listed functions support I/O software, leaving no listed function to exclude.

Primary note proof

Study-note section OS34

> A driver converts OS requests into device-specific operations. Device independence and uniform naming hide brand-specific details; error handling reports/retries supported failures.

Study-note section OS35

> Buffering temporarily holds data in transit to absorb speed/transfer-size differences.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 45 | lines 28-33

> Application I/O Interface represents the structuring techniques and interfaces for the operating system to enable I/O devices to be treated in a standard, uniform way. The actual differences lies kernel level modules called device drivers which are custom tailored to corresponding devices but show one of the standard interfaces to applications. The purpose of the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, such as the I/O system calls. Following are the characteristics of I/O

Operating System module | PDF page 47 | lines 5-9

> Buffering - Kernel I/O Subsystem maintains a memory area known as buffer that stores data while they are transferred between two devices or between a devices with an application operation. Buffering is done to cope with a speed mismatch between the producer and consumer of a data stream or to adapt between devices that have different data transfer sizes.

Operating System module | PDF page 47 | lines 17-18

> Error Handling - An operating system that uses protected memory can guard against many kinds of hardware and application errors.

Question sources

Addis Ababa University mock exam · Question 66

Question 6 | Q00165

Which hardware unit translates a process's logical/virtual addresses to physical memory addresses during execution?

A. MMU

B. IRQ

C. DMA

D. Cache

E. Relocation Register

Hint

Track when an address or library reference is resolved and the direction of translation.

Answer A: MMU

The MMU performs execution-time translation from a process's logical or virtual address to a physical RAM address. A relocation register can supply a base used in a simple translation, but it is a register rather than the complete translating unit. IRQ concerns interrupts, DMA moves data, and a cache stores reusable information. The corrected direction therefore identifies the MMU uniquely.

Why each choice is right or wrong

A - Correct: The MMU is the hardware unit performing logical-to-physical translation.

B - Incorrect: An IRQ signals an interrupt rather than translating process addresses.

C - Incorrect: DMA transfers data between devices and memory after setup.

D - Incorrect: A cache retains reusable contents; it is not the general address-translating unit.

E - Incorrect: A relocation register holds a base parameter used by translation hardware.

Primary note proof

Study-note section OS27

> Logical/virtual addresses are generated for a process; physical addresses select RAM. An MMU translates logical addresses to physical addresses, not the reverse.

Study-note section OS28

> In the standard base/limit model, base is that process's physical starting address and limit is its permitted length. Check 0≤logical_offset<limit, then physical=base+offset.

Study-note section OS34

> Polling checks readiness repeatedly; interrupts request CPU attention; DMA transfers blocks after setup without CPU work for every byte.

Study-note section OS35

> Caching retains copies to accelerate reuse.

Study-note section OS-S04

IRQ means interrupt request.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 33 | lines 6-13

The run-time mapping from virtual to physical address is done by the memory management unit (MMU) which is a hardware device. MMU uses following mechanism to convert virtual address to physical address.
• The value in the base register is added to every address generated by a user process which is treated as offset at the time it is sent to memory. For example, if the base register value is 10000, then an attempt by the user to use address location 100 will be dynamically reallocated to location 10100.
• The user program deals with virtual addresses; it never sees the real physical addresses.

Operating System module | PDF page 34 | lines 12-16

In this type of allocation, relocation-register scheme is used to protect user processes from each other, and from changing operating-system code and data. Relocation register contains value of smallest physical address whereas limit register contains range of logical addresses. Each logical address must be less than the limit register.

Operating System module | PDF page 45 | lines 10-22

The CPU hardware uses an interrupt request line wire which helps CPU to sense after executing every instruction. When the CPU checks that a controller has put a signal on the interrupt request line, the CPU saves a state, such as the current value of the instruction pointer, and jumps to the interrupt handler routine at a fixed address. The interrupt handler part determines the cause of the interrupt performs the necessary processing and executes an interrupt instruction to return the CPU to its execution state.
The basic mechanism of interrupt enables the CPU to respond to an asynchronous event, such as when a device controller becomes ready for service. Most CPUs have two interrupt request lines.
• non-maskable interrupt - Such kind of interrupts are reserved for events like unrecoverable memory errors.
• maskable interrupt - Such interrupts can be switched off by the CPU before the execution of critical instructions that must not be interrupted.

Operating System module | PDF page 47 | lines 10-11

Caching - Kernel maintains cache memory which is region of fast memory that holds copies of data. Access to the cached copy is more efficient than access to the original.

Question sources

Addis Ababa University mock exam · Question 67

Question 7 | Q00166

For holes of sizes 100, 500, 200 and 300 in that scan order and a request of size 180, which policy is better at minimizing this request's leftover space in its selected hole? Ignore search cost and future requests.

A. First fit algorithm is better

B. Best fit algorithm is better

C. Worst fit algorithm is better

D. All of the above

Hint

Apply the stated hole-selection or base-and-bound rule with its explicit convention.

Answer B: Best fit algorithm is better

Best-fit chooses the smallest hole large enough for the request: 200, leaving 20 unused units. First-fit stops at 500 and leaves 320 units; worst-fit also chooses 500. Under the stated objective, best-fit gives the smallest remainder for this allocation. This calculation says nothing about future fragmentation or overall search speed, which require a broader workload.

Why each choice is right or wrong

A - Incorrect: First-fit selects 500 in this scan order, leaving 320 rather than the minimum remainder.

B - Correct: Best-fit selects 200 and leaves only 20 units.

C - Incorrect: Worst-fit selects 500 and leaves 320 units.

D - Incorrect: The policies do not tie: only best-fit minimizes this request's leftover space.

Primary note proof

Study-note section OS28

> First-fit selects the first adequate hole; next-fit continues from the previous search position; best-fit selects the smallest adequate hole; worst-fit selects the largest.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 34 | lines 34-45

> External
> fragmentation
> Total memory space is enough to satisfy a request or to reside a
> process in it, but it is not contiguous so it cannot be used.
> 2
> Internal
> fragmentation
> Memory block assigned to process is bigger. Some portion of memory

is left unused as it cannot be used by another process.
External fragmentation can be reduced by compaction or shuffle memory contents to place all free memory together in one large block. To make compaction feasible, relocation should be dynamic

Question sources

Addis Ababa University mock exam · Question 68

Question 8 | Q00167

All jobs arrive together at submission, have known finite CPU bursts, perform no I/O, and no further jobs arrive. Assume one CPU, zero switching overhead, known fixed priorities (smaller value is higher; non-preemptive priority), a known positive RR quantum, and listed order for ties. For which listed algorithms can each job's total waiting time be computed at submission? Choose the most complete answer.

A. Shortest job first

B. Shortest remaining time

C. Priority based

D. Round Robin

E. All of the above

Hint

Construct a timeline and distinguish completion rates from individual elapsed-time measures.

Answer E: All of the above

The closed workload and explicit policy parameters determine each scheduling timeline. SJF orders known bursts, SRTF uses remaining bursts, priority uses the supplied fixed order, and RR uses its quantum and ready queue. Completion minus arrival minus burst then gives waiting time. Each named algorithm is individually computable, so the aggregate option is the most complete answer under these assumptions.

Why each choice is right or wrong

A - Incorrect: SJF waiting is computable, but this option omits the other computable policies.

B - Incorrect: SRTF waiting is computable for the known closed workload, but this is incomplete.

C - Incorrect: Fixed-priority waiting is computable with the supplied priorities and tie rule, but this is incomplete.

D - Incorrect: RR waiting is computable with the known quantum and queue order, but this is incomplete.

E - Correct: All four schedules and their total waiting times can be computed from the fully specified finite workload.

Primary note proof

Study-note section OS13

For arrival A, first start S and completion C, response=S−A and turnaround=C−A. For a single CPU burst B with no I/O, waiting=C−A−B.

Study-note section OS16

Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs. SRTF reconsiders on arrivals and selects the shortest remaining work; a shorter newcomer can preempt.

Study-note section OS17

Priority scheduling chooses the highest-priority ready job. Check whether a smaller or larger numeric value means higher priority. State whether a high-priority arrival preempts.

Study-note section OS18

RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Question sources

Addis Ababa University mock exam · Question 69

Question 9 | Q00168

One is not a problem caused by concurrency of processes

A. Race condition

B. Starvation

C. Deadlock

D. Mutual exclusion

E. None

Hint

Identify conflicting shared accesses and the safety or progress property required.

Answer D: Mutual exclusion is a desired safety property.

OS10 explicitly contrasts mutual exclusion, a desired safety property, with race conditions, starvation and deadlock as possible failures. Although a badly designed locking scheme can introduce deadlock, that does not turn the mutual-exclusion requirement itself into a concurrency failure. Therefore it is the single option that is not a problem in the sense used by the list.

Why each choice is right or wrong

A - Incorrect: A race is a concurrency failure caused by uncoordinated interleaving of conflicting accesses.

B - Incorrect: Starvation is a possible indefinite-delay failure of concurrent scheduling or resource competition.

C - Incorrect: Deadlock is a possible failure in which blocked participants cannot supply one another's required events.

D - Correct: Mutual exclusion is a correctness property protecting shared state, rather than a failure to be avoided.

E - Incorrect: Mutual exclusion provides a clear non-failure option, so none of the choices is not appropriate.

Primary note proof

Study-note section OS10

> A race occurs when results depend on uncoordinated interleaving of conflicting accesses. A critical section is code accessing shared state that needs protection, not the resource itself. A correct exclusion solution requires mutual exclusion, progress and bounded waiting under stated assumptions. Mutual exclusion is a desired safety property; races, starvation and deadlock are possible failures.

Study-note section OS22

A set is deadlocked when each member waits for an event/resource that only another blocked member can provide. Mere waiting for another holder is normal blocking; indefinite unfair delay can be starvation without a cycle of dependencies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

Question sources

Addis Ababa University mock exam · Question 70

Question 10 | Q00169

Which methods can participate in recovery after deadlock has been detected? Assume checkpoints and safe resource reclamation are available where needed. Select all that apply.

A. Kill the process

B. Rollback

C. Non-preempt the resource

D. Lock the process

Hint

Separate handling before allocation from actions taken after detection, and consider the cost of checking.

Answer A, B: Kill the process; Rollback

Terminating a selected process can release resources and break the dependency cycle. Recovery may instead reclaim safely preemptable resources and restore affected work to a checkpoint, so rollback can also participate. Merely preserving non-preemption or locking a process does not remove the deadlocked dependencies. With checkpoint and reclamation support stated, both termination and rollback are valid recovery tools.

Why each choice is right or wrong

A - Correct: Killing a selected victim can release resources and break the deadlock.

B - Correct: Rollback can restore affected work after safe resource reclamation.

C - Incorrect: Non-preemption preserves a deadlock-enabling restriction rather than recovering the processes.

D - Incorrect: Locking a blocked process does not release resources or restore progress.

Primary note proof

Study-note section OS26

> Detection asks whether the current state is deadlocked. Recovery can terminate selected processes or reclaim safely preemptable resources and roll affected work back to a checkpoint. Termination and rollback are both possible recovery tools; choose according to the resource and process conditions.

Study-note section OS22

> No resource preemption means release must be voluntary.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Addis Ababa University mock exam · Question 71

Question 11 | Q00170

One deadlock prevention method says, "if a process must wait for a needed resource, it drops all of its previously held resources and tries to acquire all resources gain." These attacks

A. Circular wait condition

B. No preemption conditions

C. Mutual exclusion condition

D. Hold and wait condition

E. None of the above

Hint

Identify the resource-request restriction and which necessary condition it removes.

Answer D: The policy removes hold-and-wait.

Under this policy, a process releases all previously held resources when it must wait. Releasing all resources before retrying removes simultaneous holding while waiting. The stem describes voluntary dropping, not forced withdrawal by another agent, so hold-and-wait is the directly attacked condition. It supplies no global ordering or resource-sharing rule.

Why each choice is right or wrong

A - Incorrect: The policy does not impose a global acquisition order, which is the direct circular-wait prevention method.

B - Incorrect: The stem specifies the process dropping resources, not a general rule permitting forced resource preemption.

C - Incorrect: Resources may remain nonshareable; releasing holdings does not remove mutual-exclusion requirements.

D - Correct: A process no longer retains allocated resources while waiting to acquire the rest.

E - Incorrect: Hold-and-wait is explicitly removed by the release-all-before-waiting rule, so an identified condition fits.

Primary note proof

Study-note section OS24

> Prevention constrains how requests are made. Share a resource where safe to remove mutual exclusion; request all needed resources before holding any to remove hold-and-wait; permit safe resource withdrawal/restart to attack no-preemption; impose a global acquisition order to remove circular wait. Not every physical resource is shareable or safely preemptable.

Study-note section OS24

> “If waiting, release all held resources and try again” removes simultaneous holding while waiting.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Addis Ababa University mock exam · Question 72

Question 12 | Q00229

Which statements are false? Interpret aging under a priority scheme and workload where increasing waiting jobs' priority prevents indefinite postponement. Select all that apply.

A. File system standards of various OS is the same.

B. SJF may cause starvation.

C. Aging is the solution of the starvation problem in the case of priority scheduling.

D. Critical section is a resource that cannot be accessed by multiple processes concurrently.

Hint

Identify conflicting shared accesses and the safety or progress property required.

Answer A, D: File system standards of various OS is the same.; Critical section is a resource that cannot be accessed by multiple processes concurrently.

Filesystems are not governed by one identical format across all operating systems; the notes distinguish FAT32, NTFS and ext4. A critical section is code that accesses shared state, not the resource itself. SJF can starve a long job when short work continues arriving. Aging addresses priority starvation under the stated policy assumptions, so those latter two statements are true.

Why each choice is right or wrong

A - Correct: The universal sameness claim is false; different filesystem formats exist.

B - Incorrect: SJF may starve long work under repeated short arrivals, so this statement is true.

C - Incorrect: Aging raises effective priority and addresses starvation under the specified conditions, so this statement is true.

D - Correct: A critical section is protected code rather than the shared resource, making this statement false.

Primary note proof

Study-note section OS10

> A critical section is code accessing shared state that needs protection, not the resource itself.

Study-note section OS17

> Fixed-priority scheduling can starve low-priority jobs under continued high-priority load. SJF can also starve long jobs; starvation is not exclusive to fixed-priority scheduling.

Study-note section OS17

> Aging increases waiting jobs' effective priority according to a rule, reducing indefinite postponement when the priority scheme and workload permit.

Study-note section OS36

> Sockets also support ordinary read/write, alongside socket/connect/send operations, so read is not uniquely "not a network call." FAT32, NTFS and ext4 are filesystem formats; HTTP is an application communication protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 48 | lines 5-8

13.1 File
A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the file's creator and user.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 10 | lines 23-28

• Loads a program into memory.
• Executes the program.
• Handles program's execution.
• Provides a mechanism for process synchronization.
• Provides a mechanism for process communication.
• Provides a mechanism for deadlock handling.

Question sources

Mock exam 4 · Question 31

Question 13 | Q00230

One of the following system calls is used to create a new process.

A. Fork()

B. exit()

C. abort()

D. exec()

Hint

Separate creation, image replacement and termination; classify termination by its cause.

Answer A: fork().

OS07 states that fork() creates a child in POSIX-style systems, returning zero in the child and its PID in the parent on success. exec replaces the current process image and does not itself create another process. exit and abort terminate execution normally and abnormally respectively. Creation therefore corresponds to fork among the literal listed calls.

Why each choice is right or wrong

A - Correct: fork creates a child process rather than replacing or terminating the existing process image.

B - Incorrect: exit terminates normally through the runtime rather than creating a new process.

C - Incorrect: abort requests abnormal termination rather than allocating a new child execution instance.

D - Incorrect: exec replaces the current process image; it does not itself create another process.

Primary note proof

Study-note section OS07

> In POSIX-style systems fork() creates a child: zero is returned in the child and the child PID in the parent on success. The two have distinct address spaces, although resources may be shared through explicit mechanisms. exec replaces the current process image; it does not itself create another process.

Study-note section OS07

> Normal completion and a program-detected error exit are voluntary; an unrecoverable fault or being killed is involuntary. exit() terminates normally through the runtime; abort() requests abnormal termination.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 19 | lines 6-12

> A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
> • The process could issue an I/O request and then it would be placed in an I/O queue.
> • The process could create new sub process and will wait for its termination.
> • The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Operating System module | PDF page 16 | lines 21-24

> A program by itself is not a process. It is a static entity made up of program statement while process is a dynamic entity. Program contains the instructions to be executed by processor. A program takes a space at single place in main memory and continues to stay there. A program does not perform any action by itself.

Question sources

Mock exam 4 · Question 32

Question 14 | Q00232

Which of the following process scheduling algorithms select the next process to be executed based on each process burst time?

A. FCFS

B. Shortest Job First

C. Round robin

D. Priority scheduling algorithm

Hint

Compare the ready jobs using the stated burst or remaining-time rule.

Answer B: Shortest Job First

SJF requires knowing how long a process will take (PDF p23); that is the burst-length selection described. Priority may use time as one possible priority input (p24), but its general rule is highest assigned priority.

Why each choice is right or wrong

A - Incorrect: FCFS orders arrival (p23), not burst length.

B - Correct: SJF chooses short work based on execution duration (p23).

C - Incorrect: RR assigns a fixed quantum (p25).

D - Incorrect: Priority selects assigned priority; time requirements are only one possible input (p24).

Primary note proof

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 23-26

> 7.2 Shortest Job First (SJF)
> • Best approach to minimize waiting time.

• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Mock exam 4 · Question 34

Question 15 | Q00233

_____ is a software solution of a critical section problem if the number of instances and processes are more than two.

A. Semaphore

B. Compare and swap

C. Peterson

D. Test set

Hint

Track available permits and blocked operations using the stated semaphore model.

Answer A: Semaphore.

OS12 explicitly says a semaphore can coordinate more than two processes, while classic Peterson's algorithm is for two under its memory-model assumptions. OS11 explains counting semaphores as tracking several permits, fitting multiple resource instances. Compare-and-swap and test-and-set are atomic primitives rather than the named software coordination abstraction requested by the stem.

Why each choice is right or wrong

A - Correct: Counting semaphores track multiple permits and coordinate more than two participating processes.

B - Incorrect: Compare-and-swap is an atomic checking/updating primitive, rather than the requested software semaphore abstraction.

C - Incorrect: Classic Peterson's algorithm solves the two-process case, not the stated more-than-two configuration.

D - Incorrect: Test-and-set is an atomic primitive; the software multiple-permit coordination mechanism here is a semaphore.

Primary note proof

Study-note section OS11

> A semaphore provides atomic wait/P/down and signal/V/up operations. Under the nonnegative-token model, wait consumes one available token or blocks when none is available; signal supplies a token and may wake a waiter. Implementations using negative internal counts encode waiters differently, so state the model. A binary semaphore represents one permit; a counting semaphore tracks several. A mutex normally has ownership rules that a general semaphore lacks.

Study-note section OS12

> Test-and-set and compare-and-swap combine checking/updating atomically; ordinary check-then-write does not. Peterson's classic algorithm uses flags plus turn for two processes under its memory-model assumptions; a semaphore can coordinate more than two.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Mock exam 4 · Question 35

Question 16 | Q00359

A process can be terminated due to:

A. killed by another process

B. normal exit

C. all of the mentioned

D. fatal error

Hint

Separate creation, image replacement and termination; classify termination by its cause.

Answer C: All listed causes can terminate a process.

OS07 distinguishes voluntary normal completion from involuntary faults or being killed. Each named cause is therefore a genuine termination cause, although the causes differ in who initiates them and whether completion is normal. The aggregate choice includes all three. Selecting only one named cause would omit other valid termination paths supported by the complete note.

Why each choice is right or wrong

A - Incorrect: Being killed can terminate a process, but this alone omits normal completion and fatal faults.

B - Incorrect: Normal exit is valid, but it is only one of the three supported termination causes.

C - Correct: Being killed, normal exit and fatal error all provide legitimate process-termination causes.

D - Incorrect: A fatal error can terminate execution, but the other two named causes are also valid.

Primary note proof

Study-note section OS07

> Normal completion and a program-detected error exit are voluntary; an unrecoverable fault or being killed is involuntary. exit() terminates normally through the runtime; abort() requests abnormal termination.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 29-39

> 1. New The process is being created.
> 2.
> Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
> 3.
> Running Process instructions are being executed (i.e. The process that is currently being

executed).
4.
Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
5. Terminated The process has finished execution.

Operating System module | PDF page 19 | lines 20-25

Non-Running Processes that are not running are kept in queue, waiting for their turn to execute. Each entry in the queue is a pointer to a particular process. Queue is implemented by using linked list. Use of dispatcher is as follows. When a process is interrupted, that process is transferred in the waiting queue. If the process has completed or aborted, the process is discarded. In either case, the dispatcher then selects a process from the queue to execute.

Question sources

Computer Science model exam · Question 90

University of Gondar model exam · Question 90

Question 17 | Q00360

Which named interface directly requests protected operating-system services across the user/kernel boundary, rather than merely providing application-facing wrappers?

A. library

B. system calls

C. assembly instructions

D. API

Hint

Identify the abstraction layer and separate a protected service request from an application task.

Answer B: system calls

A system call requests a protected kernel service, such as process creation or file access. An application may call an API implemented by a library, but that wrapper is distinct from the direct user/kernel service boundary. Assembly can encode the machine instructions used to enter that boundary without being its general interface name. The requested abstraction is therefore system calls.

Why each choice is right or wrong

A - Incorrect: A library can wrap system calls but is not itself the direct kernel-service boundary.

B - Correct: System calls directly request protected kernel services.

C - Incorrect: Assembly instructions are a programming representation, not the named service-interface abstraction.

D - Incorrect: An API is the application-facing interface and may wrap the direct system-call request.

Primary note proof

Study-note section OS04

> A system call requests a protected kernel service, such as creating a process, reading a file or opening a socket. A user-facing API/library may wrap that request; not every library function enters the kernel. Distinguish the application API from the direct user/kernel boundary when describing access to OS services.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 16-19

> implement most of the functionalities of the operating system and do not requires kernel module's code access rights.
> System Utility - System Utility programs are responsible to do specialized, individual level tasks.

Operating System module | PDF page 45 | lines 28-33

> Application I/O Interface represents the structuring techniques and interfaces for the operating system to enable I/O devices to be treated in a standard, uniform way. The actual differences lies kernel level modules called device drivers which are custom tailored to corresponding devices but show one of the standard interfaces to applications. The purpose of the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, such as the I/O system calls. Following are the characteristics of I/O

Question sources

Computer Science model exam · Question 91

Operating Systems practice questions · Question 4

University of Gondar model exam · Question 91

Question 18 | Q00361

Which module gives control of the CPU to the process selected by the short-term scheduler?

A. Interrupt

B. Dispatcher

C. Scheduler

D. none of the mentioned

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer B: Dispatcher

The module uses overlapping terminology: it calls the short-term scheduler dispatcher on PDF p20, and describes the dispatcher selecting execution on p19. Under the conventional selection-versus-handoff reading requested by the stem, B is the matching term; p21 explains the handoff state work.

Why each choice is right or wrong

A - Incorrect: Interrupts trigger handling, not the named CPU-handoff module (p45).

B - Correct: Dispatcher is the execution-transfer role described on p19, with state switching on p21.

C - Incorrect: The stem already assigns selection to the scheduler; note that p20 loosely equates it with dispatcher.

D - Incorrect: The module explicitly names a dispatcher, so none is not appropriate.

Primary note proof

Study-note section OS06

> Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work. The dispatcher performs the handoff/context restore after selection.

Study-note section OS05

> A context switch saves one execution context and restores another; useful application work is interrupted by this overhead.

Study-note section OS34

> Polling checks readiness repeatedly; interrupts request CPU attention; DMA transfers blocks after setup without CPU work for every byte.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 19 | lines 20-25

> Non-Running Processes that are not running are kept in queue, waiting for their turn to execute. Each entry in the queue is a pointer to a particular process. Queue is implemented by using linked list. Use of dispatcher is as follows. When a process is interrupted, that process is transferred in the waiting queue. If the process has completed or aborted, the process is discarded. In either case, the dispatcher then selects a process from the queue to execute.

Operating System module | PDF page 20 | lines 15-21

> It is also called CPU scheduler. Main objective is increasing system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects process among the processes that are ready to execute and allocates CPU to one of them.

Short term scheduler also known as dispatcher, execute most frequently and makes the fine-grained decision of which process to execute next. Short term scheduler is faster than long term scheduler.

Operating System module | PDF page 21 | lines 41-50

A context switch is the mechanism to store and restore the state or context of a CPU in Process Control block so that a process execution can be resumed from the same point at a later time. Using this technique, a context switcher enables multiple processes to share a single CPU. Context switching is an essential part of a multitasking operating system features.
When the scheduler switches the CPU from executing one process to execute another, the context switcher saves the content of all processor registers for the process being removed from the CPU, in its process descriptor. The context of a process is represented in the process control block of a process. Context switch time is pure overhead. Context switching can significantly affect performance as modern computers have a lot of general

Question sources

Computer Science model exam · Question 92

University of Gondar model exam · Question 92

Question 19 | Q00362

Which of the following condition is required for deadlock to be possible?

A. mutual exclusion

B. all of the mentioned

C. a process may hold allocated resources while awaiting assignment of other resources

D. no resource can be forcibly removed from a process holding it

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer B: All named conditions are necessary conditions.

OS22 lists mutual exclusion, hold-and-wait, no resource preemption and circular wait as necessary in the standard reusable-resource model. The three substantive choices name the first three, so all of the mentioned is the comprehensive option. Circular wait is also required despite being omitted from the list. Having these policies does not prove a particular current allocation is already deadlocked.

Why each choice is right or wrong

A - Incorrect: Mutual exclusion is necessary, but selecting it alone omits the other two necessary conditions offered.

B - Correct: Each substantive option names a standard necessary deadlock condition; the aggregate includes all offered conditions.

C - Incorrect: Retaining resources while requesting more is hold-and-wait, one valid condition among the three offered.

D - Incorrect: No forced removal is no-preemption, which is necessary but does not exclude the other valid conditions.

Primary note proof

Study-note section OS22

> Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Study-note section OS22

> All four are necessary in the standard reusable-resource model.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Computer Science model exam · Question 93

University of Gondar model exam · Question 93

Question 20 | Q00363

Which scheduling algorithm allocates the CPU first to the process that requests the CPU first?

A. first-come, first-served scheduling

B. priority scheduling

C. shortest job scheduling

D. none of the mentioned

Hint

Follow arrival order, include idle periods and distinguish waiting from completion time.

Answer A: first-come, first-served scheduling

Jobs execute first come, first served on PDF p23. The first CPU requester is consequently the first served in the FCFS queue.

Why each choice is right or wrong

A - Correct: Directly matches the arrival-order rule (p23).

B - Incorrect: Uses highest priority (p24).

C - Incorrect: Uses execution-length information (p23).

D - Incorrect: An offered policy already matches the rule.

Primary note proof

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.
> • Easy to understand and implement.
> • Poor in performance as average wait time is high.
> Wait time of each process is following
> Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 24 | lines 10-15

> 7.3 Priority Based Scheduling
> • Each process is assigned a priority. Process with highest priority is to be executed first and so on.
> • Processes with same priority are executed on first come first serve basis.
> • Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 23 | lines 23-26

> 7.2 Shortest Job First (SJF)
> • Best approach to minimize waiting time.
> • Impossible to implement
> • Processer should know in advance how much time process will take.

Variant 1 | Alternative version 1

Which scheduling algorithm allocates the CPU first to the process that requests the CPU first?

A. Priority scheduling

B. Shortest job scheduling

C. First-come, first-served scheduling

D. None of above

Answer C: First-come, first-served scheduling

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Incorrect: Priority scheduling chooses assigned rank, not request order (PDF p24).

B - Incorrect: Shortest-job scheduling chooses duration, not request order (PDF p23).

C - Correct: FCFS serves jobs first come, first served (PDF p23). The variant moves the correct text from current A to variant C.

D - Incorrect: Variant C supplies the arrival-order policy, so none does not apply (PDF p23).

Primary note proof

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job.

Question sources

Computer Science model exam · Question 94

University of Gondar model exam · Question 94

Computer Science model exam 2 · Question 75

Question 21 | Q00607

Which directory implementation is used in most Operating System?

A. Single level directory structure

B. Two level directory structure

C. Tree directory structure

D. Acyclic directory structure

Hint

Follow namespace structure and distinguish names, underlying files and mounted filesystems.

Answer C: Tree directory structure, in the usual general-purpose OS sense.

OS37 states that modern general-purpose systems commonly present hierarchical namespaces and defines a tree as allowing nested directories and absolute/relative paths. That directly supports the conventional tree-directory choice. This is a general-purpose namespace convention, not a measured claim about every OS ever built; links can add graph structure without replacing the hierarchy as the basic user-visible organization.

Why each choice is right or wrong

A - Incorrect: A single namespace without nested directories does not match the common hierarchical organization described.

B - Incorrect: Separating user directories alone is less general than the nested hierarchy specified for modern general-purpose systems.

C - Correct: Nested directories and paths form the tree/hierarchical organization the note identifies as commonly presented.

D - Incorrect: Links can add graph sharing, but acyclic sharing is not the basic common directory organization identified.

Primary note proof

Study-note section OS37

> A single-level directory requires unique names in one namespace. Two-level directories separate users. A tree allows nested directories and absolute/relative paths; links can create shared graphs. Modern general-purpose systems commonly present hierarchical namespaces, but a feature name alone does not prove that every system excludes simpler arrangements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 19-23

> A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Operating System does the following activities for file management.
> Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.

Operating System module | PDF page 54 | lines 5-17

> Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
> Open Source - Linux source code is freely available and it is community-based

development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
Shell - Linux provides a special interpreter program which can be used to execute

Question sources

Computer Science model exam · 2015 · Question 76

Question 22 | Q00696

In the basic virtual-memory models, which technique divides the address space into fixed-size units mapped to equal-size physical frames?

A. Paging

B. Segmentation

C. A&B

D. All

Hint

Split the address into page and offset, then apply the mapping and access checks.

Answer A: Paging

Paging divides addresses into fixed-size pages and maps them to physical frames of the same size. Segmentation instead uses variable-length logical units such as code and stack. Thus the fixed-size condition selects paging alone. Both combined alternatives include segmentation and therefore fail that condition; fixed partitions elsewhere in an operating system are outside the page-versus-segment comparison.

Why each choice is right or wrong

A - Correct: Paging uses fixed-size pages and corresponding equal-size frames.

B - Incorrect: Segments are variable-length logical units, so segmentation does not satisfy the fixed-unit condition.

C - Incorrect: Including segmentation makes this combined choice fail the fixed-size requirement.

D - Incorrect: All includes the variable-size segmentation alternative and is therefore incorrect.

Primary note proof

Study-note section OS30

Pages and physical frames have equal fixed size P.

Study-note section OS31

Segmentation uses logical variable-length units such as code, stack and arrays.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 34 | lines 47-56

External fragmentation is avoided by using paging technique. Paging is a technique in which physical memory is broken into blocks of the same size called pages (size is power of 2, between 512 bytes and 8192 bytes). When a process is to be executed, it's corresponding pages are loaded into any available memory frames.
Logical address space of a process can be non-contiguous and a process is allocated physical memory whenever the free memory frame is available. Operating system keeps track of all free frames. Operating system needs n free frames to run a program of size n pages.
Address generated by CPU is divided into
• Page number (p) -- page number is used as an index into a page table which contains base address of each page in physical memory.

Operating System module | PDF page 35 | lines 7-12

Segmentation is a technique to break memory into logical pieces where each piece represents a group of related information. For example, data segments or code segment for each process, data segment for operating system and so on. Segmentation can be implemented using or without using paging.
Unlike paging, segment is having varying sizes and thus eliminates internal fragmentation. External fragmentation still exists but to lesser extent.

Question sources

Computer Science model exam 2 · Question 37

Question 23 | Q00731

When does a page fault occur?

A. The page is present in memory.

B. Deadlock occurs.

C. The page does not present in memory.

D. Buffering occurs.

Hint

Separate a permitted nonresident access from an illegal access and trace the required handling steps.

Answer C: The page does not present in memory.

A demand-paging access to a nonresident page causes the trap (PDF p38). PDF p39 distinguishes a valid but absent page, which is brought in, from an invalid address, which terminates the process. C presupposes an actual reference to that page.

Why each choice is right or wrong

A - Incorrect: Resident page access proceeds normally (p38).

B - Incorrect: Deadlock is not the page-residency test on p38.

C - Correct: An accessed valid page absent from RAM causes demand paging (pp38-39).

D - Incorrect: Buffering is not the fault-triggering residency condition.

Primary note proof

Study-note section OS32

> Virtual memory provides translated, protected address spaces and allows nonresident portions of a program to be backed by storage. It does not add physical RAM or remove all capacity constraints. Demand paging brings a page in when accessed. On a nonresident access: trap, validate mapping/permissions, find a frame or choose a victim, save a dirty victim if needed, obtain contents, update mapping and restart.

Study-note section OS32

> Not every fault reads a disk: some allocate zero-filled pages or establish shared mappings. An illegal access is rejected, not automatically loaded.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 38 | lines 11-18

> Hardware support is required to distinguish between those pages that are in memory and those pages that are on the disk using the valid-invalid bit scheme. Where valid and invalid pages can be checked by checking the bit. Marking a page will have no effect if the process never attempts to access the page. While the process executes and accesses pages that are memory resident, execution proceeds normally.
> Access to a page marked invalid causes a page-fault trap. This trap is the result of the operating system's failure to bring the desired page into memory. But page fault can be handled as following.

Operating System module | PDF page 39 | lines 5-11

> Check an internal table for this process, to determine whether the reference was a valid or it was an invalid memory access.
> Step 2
> If the reference was invalid, terminate the process. If it was valid, but page have not yet brought in, page in the latter.
> Step 3 Find a free frame.
> Step 4 Schedule a disk operation to read the desired page into the newly allocated frame.

Question sources

Computer Science model exam 2 · Question 72

Computer Science model exam 2 · Question 90

Question 24 | Q00732

Which of the following is a condition that causes deadlock?

A. Mutual exclusion

B. Hold and wait

C. Circular wait

D. No preemption

E. All of these

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer E: All four are standard necessary deadlock conditions.

OS22 lists mutual exclusion, hold-and-wait, no-preemption and circular wait as the four necessary conditions in the standard reusable-resource model. Each substantive choice therefore names a valid deadlock condition, making all of these the comprehensive answer. The wording causes must not be read as saying one condition alone is sufficient; actual allocation/request dependencies still require inspection.

Why each choice is right or wrong

A - Incorrect: Mutual exclusion is one necessary condition, but the other three named conditions also belong to the standard set.

B - Incorrect: Hold-and-wait is necessary, but selecting it alone leaves out the other valid conditions.

C - Incorrect: Circular wait is part of the condition set, but it is not the sole valid listed condition.

D - Incorrect: No-preemption is necessary in this model, alongside the other three substantive choices.

E - Correct: All four substantive choices are standard necessary conditions; together their names give the complete condition set.

Primary note proof

Study-note section OS22

> Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Study-note section OS22

All four are necessary in the standard reusable-resource model. Their availability as general system policies does not prove a deadlock exists at this instant; inspect actual allocations/requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

Question sources

Computer Science model exam 2 · Question 73

Question 25 | Q00733

What is an operating system?

A. System service provider to the application programs

B. Collection of programs that manages hardware resources

C. Interface between the hardware and application programs

D. All of above

Hint

Separate broad resource-management goals from individual services or applications.

Answer D: All of above

The OS manages hardware and mediates users/program execution (PDF p5), and provides services to programs (p10). Thus each of A-C is a valid aspect of the definition and D collects them.

Why each choice is right or wrong

A - Incorrect: Program services are explicitly provided (p10).

B - Incorrect: Hardware and resource management are explicit (p5).

C - Incorrect: The mediation/interface role is explicit (p5), including program execution.

D - Correct: All three descriptions agree with pp5 and 10.

Primary note proof

Study-note section OS01

An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process. Application development tools use OS services but application development itself is not a core resource-management function.

Study-note section OS04

A system call requests a protected kernel service, such as creating a process, reading a file or opening a socket.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 5 | lines 6-13

An operating System (OS) is an intermediary between users and computer hardware. It provides users an environment in which a user can execute programs conveniently and efficiently. In technical terms, it is software which manages hardware. An operating System controls the allocation of resources and services such as memory, processors, devices and information.
1.1 Definition
An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Operating System module | PDF page 10 | lines 5-15

An Operating System provides services to both the users and to the programs.
• It provides programs, an environment to execute.
• It provides users, services to execute the programs in a convenient manner.
Operating Systems can provide the following common services:
✓ Program execution
✓ I/O operations
✓ File System manipulation
✓ Communication
✓ Error Detection
✓ Resource Allocation
✓ Protection

Question sources

Computer Science model exam 2 · Question 74

Computer Science model exam 2 · Question 36

Operating Systems practice questions · Question 1

Question 26 | Q00735

What is the ready state of a process?

A. When process is unable to run until some task has been completed

B. When process is scheduled to run after some execution

C. When process is using the CPU

D. None of the mentioned

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer B: When process is scheduled to run after some execution

Ready means waiting to be assigned a processor (PDF p16). B is the nearest paraphrase, provided scheduled to run means eligible and waiting, not already executing.

Why each choice is right or wrong

A - Incorrect: Waiting for an event is the Waiting state.

B - Correct: Eligible to receive CPU next matches Ready under the stated reading.

C - Incorrect: Using the CPU defines Running.

D - Incorrect: Ready is represented by B under the stated interpretation.

Primary note proof

Study-note section OS06

> New processes are created/admitted; ready means eligible for CPU but not currently running. Running uses a CPU. Waiting/blocked means unable to proceed until an event; terminated means execution has ended. Running→ready follows time-slice expiry or another preemption. Running→waiting follows blocking I/O. I/O completion normally gives waiting→ready, not a guaranteed immediate running state.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 29-39

> 1. New The process is being created.
> 2.
> Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
> 3.
> Running Process instructions are being executed (i.e. The process that is currently being executed).
> 4.
> Waiting The process is waiting for some event to occur (such as the completion of an I/O

operation).
5. Terminated The process has finished execution.

Question sources

Computer Science model exam 2 · Question 76

Question 27 | Q00736

In the standard reusable-resource model, a set of processes is deadlocked when _____.

A. Each process is terminated

B. All processes are trying to kill each other

C. Each process is blocked waiting for an event or resource that only another blocked member of the set can provide

D. None of the mentioned

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer C: Each process is blocked waiting for an event or resource that only another blocked member of the set can provide

Deadlock requires blocked participants whose needed event or resource can be supplied only by another blocked participant in the same set. The closed dependency prevents any member from progressing without intervention. Terminated processes are no longer waiting, and attempts to kill one another do not establish a resource dependency. The revised blocking option states the necessary relationship rather than merely predicting permanent delay.

Why each choice is right or wrong

A - Incorrect: Termination ends execution and does not define a set of blocked dependent processes.

B - Incorrect: Attempts to kill other processes do not establish deadlocked resource waits.

C - Correct: This states the closed dependency among blocked members that defines deadlock.

D - Incorrect: The revised C explicitly gives the deadlock definition, so none is false.

Primary note proof

Study-note section OS22

A set is deadlocked when each member waits for an event/resource that only another blocked member can provide. Mere waiting for another holder is normal blocking; indefinite unfair delay can be starvation without a cycle of dependencies.

Study-note section OS07

Normal completion and a program-detected error exit are voluntary; an unrecoverable fault or being killed is involuntary. exit() terminates normally through the runtime; abort() requests abnormal termination.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 29-39

1. New The process is being created.
2. Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
3. Running Process instructions are being executed (i.e. The process that is currently being executed).
4. Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
5. Terminated The process has finished execution.

Operating System module | PDF page 10 | lines 23-28

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

Question sources

Computer Science model exam 2 · Question 77

Question 28 | Q00737

In Operating Systems, which of the following is/are CPU scheduling algorithms?

A. Priority

B. Round Robin

C. Shortest Job First

D. All of the mentioned

Hint

Compare the ready jobs using the stated burst or remaining-time rule.

Answer D: All of the mentioned

PDF pp23-25 explicitly cover priority, RR and SJF in the scheduling chapter, so D collects three genuine CPU scheduling policies.

Why each choice is right or wrong

A - Incorrect: Priority ordering is defined on p24.

B - Incorrect: Quantum-based scheduling is defined on p25.

C - Incorrect: Duration-based SJF is defined on p23.

D - Correct: All three belong to the chapter's scheduling methods.

Primary note proof

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.
> • Easy to understand and implement.
> • Poor in performance as average wait time is high.
> Wait time of each process is following
> Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

> 7.2 Shortest Job First (SJF)
> • Best approach to minimize waiting time.
> • Impossible to implement
> • Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

> 7.3 Priority Based Scheduling
> • Each process is assigned a priority. Process with highest priority is to be executed first and so on.
> • Processes with same priority are executed on first come first serve basis.
> • Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Question sources

Computer Science model exam 2 · Question 78

Operating Systems practice questions · Question 3

Question 29 | Q00738

In which of the storage placement strategies a program is placed in the largest available hole in the main memory?

A. Best fit

B. First fit

C. Worst fit

D. Buddy

Hint

Apply the stated hole-selection or base-and-bound rule with its explicit convention.

Answer C: Worst-fit.

OS28 defines worst-fit as selecting the largest hole, directly matching the stem. Best-fit chooses the smallest adequate hole, and first-fit stops at the first adequate one rather than comparing all sizes. The buddy label does not state this largest-hole rule in the provided note. No numeric trace is necessary because the selection criterion itself is explicit.

Why each choice is right or wrong

A - Incorrect: Best-fit chooses the smallest adequate hole, which is the opposite size-selection criterion from the stem.

B - Incorrect: First-fit selects the first adequate hole in search order, not necessarily the largest available hole.

C - Correct: Worst-fit is explicitly the policy that selects the largest available adequate hole.

D - Incorrect: The supplied largest-hole criterion is defined as worst-fit; the buddy label is not that rule.

Primary note proof

Study-note section OS28

First-fit selects the first adequate hole; next-fit continues from the previous search position; best-fit selects the smallest adequate hole; worst-fit selects the largest.

Study-note section OS-S03

A buddy allocator manages power-of-two block sizes. It takes a smallest sufficient block, splitting larger blocks when necessary, and can merge a freed block with its free buddy. This is not the worst-fit rule of selecting the largest arbitrary hole.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 34 | lines 34-45

External
fragmentation
Total memory space is enough to satisfy a request or to reside a
process in it, but it is not contiguous so it cannot be used.
2
Internal
fragmentation
Memory block assigned to process is bigger. Some portion of memory
is left unused as it cannot be used by another process.
External fragmentation can be reduced by compaction or shuffle memory contents to place all free memory together in one large block. To make compaction feasible, relocation should be dynamic

Question sources

Computer Science model exam 2 · Question 79

Question 30 | Q00916

Which one of the following function of operating system is categorized under process management?

A. Allocates the device in the efficient way

B. De-allocates the resources.

C. .
Allocates the memory when the process requests it to do so.

D. Allocates the processor (CPU) to a process

Hint

Separate broad resource-management goals from individual services or applications.

Answer D: Allocates the processor (CPU) to a process

CPU allocation is explicitly process/processor management (PDF p6). Memory requests belong to memory management (p5), and device allocation belongs to device management (p6).

Why each choice is right or wrong

A - Incorrect: Device allocation is listed under device management.

B - Incorrect: Resource deallocation is too broad and appears in several management categories, not this precise processor task.

C - Incorrect: Memory allocation is explicitly under memory management (p5).

D - Correct: Allocating CPU to a process is explicitly listed under processor management (p6).

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs.

Study-note section OS06

> Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 4-9

> In multiprogramming environment, OS decides which process gets the processor when and how much time. This function is called process scheduling. Operating System does the following activities for processor management. Keeps tracks of processor and status of process. Program responsible for this task is known as traffic controller.
> Allocates the processor (CPU) to a process.
> De-allocates processor when processor is no longer required.

Operating System module | PDF page 5 | lines 26-36

> Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address. Main memory provides a fast storage that can be access directly by the CPU. So, for a program to be executed, must in the main memory.
> Activities Operating System for memory management.
> Keeps tracks of primary memory (i.e. what part of it are in use by whom, what part are not in use.)
> In multiprogramming, OS decides which process will get memory when and how much.
> Allocates the memory when the process requests it to do so.
> De-allocates the memory when the process no longer needs it or has been terminated.

Operating System module | PDF page 6 | lines 11-17

> OS manages device communication via their respective drivers. Activities of Operating System for device management.
> Keeps tracks of all devices. Program responsible for this task is known as the I/O

controller.
Decides which process gets the device when and for how much time.
Allocates the device in the efficient way.
De-allocates devices.

Question sources

Computer Science Exit Exam · 2018 · Question 29

Question 31 | Q00928

What is the primary function of an operating system?

A. To connect to the internet

B. To create documents and spreadsheets

C. .
To design computer hardware

D. To act as an intermediary between users and computer hardware

Hint

Separate broad resource-management goals from individual services or applications.

Answer D: To act as an intermediary between users and computer hardware

PDF p5 defines the OS as intermediary between users and hardware and controller of program execution. D states this general role; the other options are particular applications or other activities.

Why each choice is right or wrong

A - Incorrect: Internet connection is narrower than the whole OS definition.

B - Incorrect: Document authoring is a program activity, not the definition on p5.

C - Incorrect: Hardware design is not managing hardware during execution.

D - Correct: Directly matches the intermediary definition on p5.

Primary note proof

Study-note section OS01

An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process. Application development tools use OS services but application development itself is not a core resource-management function.

Study-note section OS03

A shell reads/interprets user commands; a GUI is another interface. Browsers, office tools and editors are applications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 5 | lines 6-13

An operating System (OS) is an intermediary between users and computer hardware. It provides users an environment in which a user can execute programs conveniently and efficiently. In technical terms, it is software which manages hardware. An operating System controls the allocation of resources and services such as memory, processors, devices and information.
1.1 Definition
An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Question sources

Computer Science Exit Exam · 2018 · Question 41

Question 32 | Q00935

A user installs a new printer. The OS automatically detects and configures it. This is an example of:

A. Plug-and-play functionality

B. Manual driver installation

C. .
Manual hardware management

D. Resource allocation

Hint

Separate device hardware, device-specific software and general event or transfer mechanisms.

Answer A: Plug-and-play functionality.

OS34 defines plug-and-play as discovering and configuring compatible new hardware, with the qualification that a driver may still be required. That is exactly the printer scenario. The automatic wording excludes the two manual alternatives. Resource allocation may occur during configuration, but it is broader than the specific automatic detection/configuration feature requested by the question.

Why each choice is right or wrong

A - Correct: Automatically discovering and configuring the installed printer is precisely the plug-and-play behavior described.

B - Incorrect: The scenario explicitly says automatic detection/configuration rather than manual driver installation by the user.

C - Incorrect: Manual hardware management conflicts with the automatic configuration behavior specified in the scenario.

D - Incorrect: Resources may be assigned during setup, but the specific automatic discovery feature is plug-and-play.

Primary note proof

Study-note section OS34

> Plug-and-play discovers/configures compatible new hardware but may still require a driver.

Study-note section OS34

> A driver converts OS requests into device-specific operations. Device independence and uniform naming hide brand-specific details; error handling reports/retries supported failures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 30-37

> I/O subsystem comprised of I/O devices and their corresponding driver software. Drivers hides the peculiarities of specific hardware devices from the user as the device driver knows the peculiarities of the specific device.
> Operating System manages the communication between user and device drivers.
> The major activities of an operating system with respect to I/O Operation.
> • I/O operation means read or write operation with any file or any specific I/O device.
> • Program may require any I/O device while running.
> • Operating system provides the access to the required I/O device when required.

Operating System module | PDF page 47 | lines 19-26

> 12.5 Device driver
> Device driver is a program or routine developed for an I/O device. A device driver implements I/O operations or behaviors on a specific class of devices. For example a system supports one or a number of multiple brands of terminals, all slightly different terminals may have a single terminal driver. In the layered structure of I/O system, device driver lies between interrupt handler and device independent I/O software. The job of a device driver is following.
> To accept request from the device independent software above it.
> To see to it that the request is executed.

Question sources

Computer Science Exit Exam · 2018 · Question 48

Question 33 | Q00944

In asystem, resources cannot be forcibly taken from processes; they must be released voluntarily. Which deadlock condition does this represent?

A. No Preemption

B. Hold and Wait

C. .
Mutual Exclusion

D. Circular Wait

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer A: No preemption.

OS22 directly defines no resource preemption as requiring voluntary release. The stem repeats that resources cannot be forcibly taken, so the match is exact. Hold-and-wait concerns retaining one resource while requesting another, mutual exclusion concerns exclusive possession, and circular wait concerns a closed dependency chain. None of those separately states the prohibition on forced withdrawal.

Why each choice is right or wrong

A - Correct: Prohibiting forced withdrawal and requiring voluntary release is exactly the no-preemption resource condition.

B - Incorrect: Hold-and-wait concerns retaining a resource while requesting another, not whether forced removal is permitted.

C - Incorrect: Mutual exclusion limits concurrent holders; it does not itself require voluntary release.

D - Incorrect: Circular wait describes a closed chain of dependencies rather than the resource-withdrawal rule.

Primary note proof

Study-note section OS22

> Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.

- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

Question sources

Computer Science Exit Exam · 2018 · Question 57

Question 34 | Q00984

How does an operating system improve efficiency in a multi-user environment?

A. By allowing only one user to access the system at a time

B. By disabling all background services

C. .
By using time-sharing and process scheduling to allocate CPU time fairly among users and processes

D. By requiring manual intervention for each task

Hint

Compare interaction, CPU sharing and deadline requirements under the stated system model.

Answer C: .
By using time-sharing and process scheduling to allocate CPU time fairly among users and processes

Time-sharing switches CPU among users and gives each a small portion of time (PDF p7), reducing idle time and giving quick response. C captures that scheduling-based allocation.

Why each choice is right or wrong

A - Incorrect: One user at a time contradicts the simultaneous time-sharing service on p7.

B - Incorrect: Disabling background services is not the mechanism described.

C - Correct: CPU scheduling and time-sharing are explicitly the mechanism on p7.

D - Incorrect: Manual intervention per task is not the CPU-switching mechanism.

Primary note proof

Study-note section OS02

Batch jobs run with little interaction after submission. Multiprogramming keeps several jobs available so another can use the CPU while one waits; it does not require fixed time slices. Time-sharing uses rapid scheduling/preemption to support interactive users. A single CPU interleaves execution; multiple CPUs can execute work truly in parallel. Shared-memory multiprocessors do not require each CPU to have private main memory.

Study-note section OS20

Fair-share scheduling allocates according to stated weights, potentially per user/group rather than per process.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 7 | lines 20-33

Time sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multi-programmed Batch Systems and Time-Sharing Systems is that in case of multi-programmed batch systems, objective is to maximize processor use, whereas in Time-Sharing Systems objective is to minimize response time.
Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, processor execute each user program in a short burst or quantum of computation. That is if n users are present, each user can get time quantum. When the user submits the command, the response time is in few seconds at most.
Operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems

Question sources

Computer Science Exit Exam · 2018 · Question 97

Question 35 | Q00994

Which exits are involuntary? Here “Terminate exit” means an externally imposed kill, “Fatal exit” means an unrecoverable execution fault, and “Error exit” means the program chooses to exit after detecting bad input. Select all that apply.

A. Normal exit

B. Terminate exit

C. Fatal exit

D. Error exit

Hint

Separate creation, image replacement and termination; classify termination by its cause.

Answer B, C: Terminate exit; Fatal exit

An externally imposed kill and an unrecoverable execution fault stop a process involuntarily. Normal completion is chosen by the program, and detecting an input error then exiting is also a voluntary program decision. The meanings separate causes of termination instead of treating every use of the word terminate as one category. Thus the external-kill and fatal-fault choices are selected.

Why each choice is right or wrong

A - Incorrect: Normal completion is a voluntary exit.

B - Correct: The defined external kill imposes termination and is involuntary.

C - Correct: An unrecoverable execution fault imposes termination and is involuntary.

D - Incorrect: The program-detected error exit is a voluntary decision under the stated meaning.

Primary note proof

Study-note section OS07

> Normal completion and a program-detected error exit are voluntary; an unrecoverable fault or being killed is involuntary. exit() terminates normally through the runtime; abort() requests abnormal termination.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 29-39

> 1. New The process is being created.
> 2.
> Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
> 3.
> Running Process instructions are being executed (i.e. The process that is currently being executed).
> 4.
> Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
> 5. Terminated The process has finished execution.

Operating System module | PDF page 39 | lines 5-11

> Check an internal table for this process, to determine whether the reference was a valid or it was an invalid memory access.
> Step 2
> If the reference was invalid, terminate the process. If it was valid, but page have not yet brought in, page in the latter.
> Step 3 Find a free frame.
> Step 4 Schedule a disk operation to read the desired page into the newly allocated frame.

Question sources

National Exit Exam practice questions · Question 8

Computer Science Exit Exam practice collection · Question 217

Question 36 | Q01085

How do we define the term Thread?

A. Device that controls input

B. Variable that controls movement

C. Controlled execution of applications

D. None of the above

Hint

Compare per-thread execution context with shared process resources and the stated thread mapping.

Answer D: None of the above

A thread is a separate flow of execution through process code with its own PC, registers and stack (PDF p27). None of A-C states that definition; controlled execution of applications is too broad to identify a thread.

Why each choice is right or wrong

A - Incorrect: A device is not the defined flow of execution.

B - Incorrect: A variable is not a flow of control with a stack and PC.

C - Incorrect: Generic controlled application execution does not distinguish thread from process or scheduler.

D - Correct: None of A-C gives the thread definition on p27.

Primary note proof

Study-note section OS08

> A thread has its own PC, registers and stack. Threads in a process share code, address space and usually open-file resources. A process can contain many threads; a thread belongs to one process at a time. Threads are not a group of processes and need not execute sequentially. Independent processes can exist; cooperating ones communicate/share data.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 27 | lines 6-14

> A thread is a flow of execution through the process code, with its own program counter, system registers and stack. A thread is also called a light weight process. Threads provide a

way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.
Each thread belongs to exactly one process and no thread can exist outside a process. Each thread represents a separate flow of control. Threads have been successfully used in implementing network servers and web server. They also provide a suitable foundation for parallel execution of applications on shared memory multiprocessors. Following figure shows

Question sources

Web Programming practice questions · Question 18

Question 37 | Q01186

Where are the list of processes that are prepared to be executed and waiting placed?

A. Job queue

B. Ready queue

C. Execution queue

D. Process queue

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer B: Ready queue

PDF p19 says processes wait in the ready queue for CPU allocation. That is exactly prepared to execute but awaiting a processor.

Why each choice is right or wrong

A - Incorrect: Job queue contains all processes in the system (p18), rather than specifically ready ones.

B - Correct: Ready queue is the explicit CPU-waiting queue (p19).

C - Incorrect: No separate execution queue is defined for this purpose.

D - Incorrect: Process queue is generic and not the specific named queue on p19.

Primary note proof

Study-note section OS06

The ready queue contains eligible work; device queues hold blocked requests. The broad textbook job queue contains processes in the system, while an admission queue more narrowly holds jobs awaiting memory/admission. Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 19 | lines 6-12

> A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
> • The process could issue an I/O request and then it would be placed in an I/O queue.
> • The process could create new sub process and will wait for its termination.
> • The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Operating System module | PDF page 18 | lines 13-18

> The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.
> Process scheduling is an essential part of a Multiprogramming operating system. Such operating systems allow more than one process to be loaded into the executable memory at a time and loaded process shares the CPU using time multiplexing.

Question sources

Computer Science model exam 2 · Question 7

Question 38 | Q01192

CPU scheduling is the basis of _____

A. Multiprogramming operating systems

B. Larger memory-sized systems

C. Multiprocessor systems

D. None of the mentioned

Hint

Compare interaction, CPU sharing and deadline requirements under the stated system model.

Answer A: Multiprogramming operating systems

Process scheduling is explicitly essential to a multiprogramming OS (PDF p18), enabling loaded processes to time-share one CPU; multiple processors or unusually large RAM are not required.

Why each choice is right or wrong

A - Correct: Multiprogramming explicitly needs this CPU sharing (pp13,18).

B - Incorrect: Large RAM is not the defining scheduling basis.

C - Incorrect: Multiprogramming can assume a single shared processor (p13).

D - Incorrect: The module explicitly names multiprogramming.

Primary note proof

Study-note section OS02

> Multiprogramming keeps several jobs available so another can use the CPU while one waits; it does not require fixed time slices. Time-sharing uses rapid scheduling/preemption to support interactive users. A single CPU interleaves execution; multiple CPUs can execute work truly in parallel.

Study-note section OS06

> Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 18 | lines 13-18

> The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.
> Process scheduling is an essential part of a Multiprogramming operating system. Such operating systems allow more than one process to be loaded into the executable memory at a time and loaded process shares the CPU using time multiplexing.

Operating System module | PDF page 13 | lines 21-25

> 4.3 Multiprogramming
> When two or more programs are residing in memory at the same time, then sharing the processor is referred to the multiprogramming. Multiprogramming assumes a single shared processor. Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has one to execute.

Question sources

Computer Science model exam 2 · Question 13

Operating Systems practice questions · Question 5

Question 39 | Q01251

An OS designer must choose between checking for deadlock after every resource request and checking periodically. No detection cost, request rate or acceptable delay is specified. Which conclusion about a universally best checking schedule is justified?

A. Both checking on every request and checking periodically are always equally optimal

B. Checking at fixed time intervals is always optimal

C. Checking on every resource request is always optimal

D. Neither schedule is universally best; the choice trades detection overhead against delay

Hint

Separate handling before allocation from actions taken after detection, and consider the cost of checking.

Answer D: Neither schedule is universally best; the choice trades detection overhead against delay

Checking after every request can discover a problem promptly but incurs repeated detection overhead. Periodic checking reduces check frequency while allowing a deadlock to persist until the next check. Which tradeoff is suitable depends on workload and acceptable detection delay. Without those criteria neither schedule is universally optimal, nor is there a basis for claiming they are always equally good.

Why each choice is right or wrong

A - Incorrect: The two policies need not have equal overhead or detection delay.

B - Incorrect: Fixed intervals may suit some workloads, but no universal optimum follows.

C - Incorrect: Per-request checks can impose excessive overhead despite earlier detection.

D - Correct: This states the overhead-versus-delay tradeoff without asserting an unsupported universal optimum.

Primary note proof

Study-note section OS26

> Checking on every request versus periodically trades overhead against detection delay; no universal best interval exists.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Computer Science model exam 2 · Question 72

Question 40 | Q01265

Which listed operation is directly an OS/tool task? Here “bad-block recovery” specifically means autonomous drive-internal sector remapping, and “booting from disk” means the initial power-on stage before the kernel has been loaded.

A. bad-block recovery

B. booting from disk

C. disk initialization

D. all of the mentioned

Hint

Identify whether data is in transit, retained for reuse or queued for a device.

Answer C: disk initialization

Disk initialization and filesystem creation are identified as OS/tool work. Firmware starts the boot sequence before the kernel takes control, while a drive may use its own firmware to remap defective sectors. The question asks about that direct responsibility boundary, rather than broad coordination of every disk-related activity. Consequently disk initialization is the only matching choice, and the aggregate is too broad.

Why each choice is right or wrong

A - Incorrect: The specified bad-sector remapping is drive-firmware work, not the requested OS/tool operation.

B - Incorrect: The stage beginning boot is firmware work before kernel control.

C - Correct: Disk initialization/filesystem preparation is OS/tool work.

D - Incorrect: The other operations include firmware responsibilities, so all is too broad.

Primary note proof

Study-note section OS35

> Disk initialization/filesystem creation is OS/tool work, firmware begins boot, and drive firmware may remap bad sectors.

Study-note section OS03

> Firmware/bootloader stages execute before transferring control to a kernel loaded into RAM.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 11-17

> OS manages device communication via their respective drivers. Activities of Operating System for device management.
> Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.

Decides which process gets the device when and for how much time.
Allocates the device in the efficient way.
De-allocates devices.

Operating System module | PDF page 47 | lines 17-18

Error Handling - An operating system that uses protected memory can guard against many kinds of hardware and application errors.

Question sources

Computer Science model exam 2 · Question 86

Question 41 | Q01274

Which method is used to prevent threads or processes from accessing a single resource?

A. PCB

B. Semaphore

C. Job Scheduler

D. Non-Contiguous Memory Allocation

Hint

Track available permits and blocked operations using the stated semaphore model.

Answer B: A semaphore can enforce one-at-a-time access.

OS11 describes a binary semaphore as one permit and gives wait(m); critical work; signal(m) with m initialized to 1. This prevents competing threads/processes from entering the protected resource operation simultaneously; it does not mean nobody can ever use the resource. A PCB stores process state, a scheduler selects jobs, and allocation layout does not enforce this resource-access rule.

Why each choice is right or wrong

A - Incorrect: A PCB records execution and resource state rather than supplying atomic mutual-exclusion operations.

B - Correct: A one-permit semaphore with atomic wait/signal can serialize entry to the protected resource operation.

C - Incorrect: A job scheduler selects admitted or ready work; it does not by itself protect shared resource accesses.

D - Incorrect: Noncontiguous memory placement changes allocation layout rather than synchronizing access to one shared resource.

Primary note proof

Study-note section OS11

A semaphore provides atomic wait/P/down and signal/V/up operations. Under the nonnegative-token model, wait consumes one available token or blocks when none is available; signal supplies a token and may wake a waiter. Implementations using negative internal counts encode waiters differently, so state the model. A binary semaphore represents one permit; a counting semaphore tracks several. A mutex normally has ownership rules that a general semaphore lacks.

Study-note section OS11

Initialize a mutex semaphore to 1. Each thread does wait(m); critical work; signal(m). For a buffer of N slots use empty=N, full=0 and mutex=1: producer waits empty then mutex, inserts, unlocks then signals full. Waiting for empty while already holding mutex can prevent the consumer from freeing a slot.

Study-note section OS05

A saved PC identifies where execution resumes.

Study-note section OS06

Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

• Loads a program into memory.
• Executes the program.
• Handles program's execution.
• Provides a mechanism for process synchronization.
• Provides a mechanism for process communication.
• Provides a mechanism for deadlock handling.

Operating System module | PDF page 17 | lines 4-7

Each process is represented in the operating system by a process control block (PCB) also called a task control block. PCB is the data structure used by the operating system. Operating system groups all information that needs about particular process. PCB contains many pieces of information associated with a specific process which is described below.

Operating System module | PDF page 20 | lines 15-21

It is also called CPU scheduler. Main objective is increasing system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects process among the processes that are ready to execute and allocates CPU to one of them.
Short term scheduler also known as dispatcher, execute most frequently and makes the

fine-grained decision of which process to execute next. Short term scheduler is faster than long term scheduler.

Operating System module | PDF page 34 | lines 47-56

External fragmentation is avoided by using paging technique. Paging is a technique in which physical memory is broken into blocks of the same size called pages (size is power of 2, between 512 bytes and 8192 bytes). When a process is to be executed, it's corresponding pages are loaded into any available memory frames.
Logical address space of a process can be non-contiguous and a process is allocated physical memory whenever the free memory frame is available. Operating system keeps track of all free frames. Operating system needs n free frames to run a program of size n pages.
Address generated by CPU is divided into
• Page number (p) -- page number is used as an index into a page table which contains base address of each page in physical memory.

Question sources

Computer Science model exam 2 · Question 95

Question 42 | Q01283

Which of the following is not a type of operating system?

A. Windows

B. Linux

C. macOS

D. Microsoft Office

Hint

Use the stated historical or platform scope and distinguish origin from technical capability.

Answer D: Microsoft Office is an application suite.

OS40 explicitly identifies Microsoft Office as an application suite and discusses Windows, Linux and macOS as OS/kernel families. Office runs using OS services rather than providing the core resource-management platform. Thus Office is the exclusion in this list. The kernel-versus-distribution distinction for Linux does not turn it into an office application or invalidate the conventional OS-family contrast.

Why each choice is right or wrong

A - Incorrect: Windows is identified as the Microsoft operating-system family, so it is not the application-suite exclusion.

B - Incorrect: Linux names a kernel used with distributions as operating systems, unlike the Office application suite.

C - Incorrect: macOS is an operating-system product rather than a user application suite in this classification.

D - Correct: Microsoft Office is explicitly described as an application suite, not an operating system.

Primary note proof

Study-note section OS40

> Linux is a Unix-compatible kernel written from scratch by Linus Torvalds; a distribution combines it with userland, libraries and tools. Android's base kernel layer is Linux; its framework and applications are higher layers. AOSP is open source, while a shipped device can include proprietary applications/services. Windows and macOS are both proprietary products, although macOS contains open-source components, so "which is not open source?" offering both is ambiguous. Microsoft Office is an application suite.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 8 | lines 21-26

> Network Operating System runs on a server and provides server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks. Examples of network operating systems are Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.

Operating System module | PDF page 53 | lines 5-7

> Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

Question sources

Operating Systems practice questions · Question 1

Question 43 | Q01284

What is the main function of an operating system?

A. To manage hardware resources

B. To manage software applications

C. To manage user accounts

D. To manage network connections

Hint

Separate broad resource-management goals from individual services or applications.

Answer A: To manage hardware resources

Hardware resource management is the general technical definition on PDF p5. Applications, accounts and connections are narrower services and cannot replace that umbrella definition.

Why each choice is right or wrong

A - Correct: Matches the hardware-management definition (p5).

B - Incorrect: Application execution is a service within that broader role.

C - Incorrect: Accounts are one aspect of network/user administration (p8).

D - Incorrect: Network connection management is narrower than OS-wide hardware management (p8).

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process. Application development tools use OS services but application development itself is not a core resource-management function.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 5 | lines 6-13

> An operating System (OS) is an intermediary between users and computer hardware. It provides users an environment in which a user can execute programs conveniently and efficiently. In technical terms, it is software which manages hardware. An operating System controls the allocation of resources and services such as memory, processors, devices and information.
> 1.1 Definition
> An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Operating System module | PDF page 8 | lines 21-26

> Network Operating System runs on a server and provides server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks. Examples of network operating systems are Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.

Question sources

Operating Systems practice questions · Question 2

Question 44 | Q01285

Which of the following is not a memory management technique used in operating systems?

A. Paging

B. Segmentation

C. Swapping

D. Virtualization

Hint

Identify the abstraction layer and separate a protected service request from an application task.

Answer D: Virtualization, in the virtual-machine sense.

OS04 defines virtualization as presenting virtual machine environments and distinguishes it from virtual memory. OS27 describes swapping, OS30 paging and OS31 segmentation as memory-management mechanisms. Under the guest-machine sense used here, virtualization is therefore the option outside that list. The answer depends on that stated sense; the broader phrase memory virtualization would describe a different use of the word.

Why each choice is right or wrong

A - Incorrect: Paging maps fixed-size virtual pages to physical frames and is explicitly a memory-management mechanism.

B - Incorrect: Segmentation maps variable-length logical regions and is explicitly a memory-management mechanism.

C - Incorrect: Swapping moves a process to backing store and is a memory-management operation described here.

D - Correct: In the note's guest-machine sense, virtualization presents virtual computers rather than naming these process-memory techniques.

Primary note proof

Study-note section OS04

> Virtualization presents a virtual machine environment, often allowing guest operating systems on one host through a hypervisor. It differs from virtual memory, which translates a process's addresses. Kernel, shell, API and VM are not interchangeable labels.

Study-note section OS27

> Swapping traditionally moves a whole process to backing store; paging moves individual pages.

Study-note section OS30

> Pages and physical frames have equal fixed size P. Divide virtual address v into page=floor(v/P) and offset=v mod P. The page table maps a page to a frame and protection/status flags.

Study-note section OS31

Segmentation uses logical variable-length units such as code, stack and arrays.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 34 | lines 47-56

External fragmentation is avoided by using paging technique. Paging is a technique in which physical memory is broken into blocks of the same size called pages (size is power of 2, between 512 bytes and 8192 bytes). When a process is to be executed, it's corresponding pages are loaded into any available memory frames.
Logical address space of a process can be non-contiguous and a process is allocated physical memory whenever the free memory frame is available. Operating system keeps track of all free frames. Operating system needs n free frames to run a program of size n pages.
Address generated by CPU is divided into
• Page number (p) -- page number is used as an index into a page table which contains base address of each page in physical memory.

Operating System module | PDF page 35 | lines 7-12

Segmentation is a technique to break memory into logical pieces where each piece represents a group of related information. For example, data segments or code segment for each process, data segment for operating system and so on. Segmentation can be implemented using or without using paging.
Unlike paging, segment is having varying sizes and thus eliminates internal fragmentation. External fragmentation still exists but to lesser extent.

Operating System module | PDF page 33 | lines 15-19

Swapping is a mechanism in which a process can be swapped temporarily out of main memory to a backing store, and then brought back into memory for continued execution.
Backing store is a usually a hard disk drive or any other secondary storage which fast in access and large enough to accommodate copies of all memory images for all users. It must be capable of providing direct access to these memory images.

Operating System module | PDF page 37 | lines 6-11

Virtual memory is a technique that allows the execution of processes which are not completely available in memory. The main visible advantage of this scheme is that programs can be larger than physical memory. Virtual memory is the separation of user logical memory from physical memory.
This separation allows an extremely large virtual memory to be provided for programmers when only a smaller physical memory is available. Following are the situations, when entire

Question sources

Operating Systems practice questions · Question 3

Question 45 | Q01286

Which of the following is not a process scheduling algorithm?

A. First-Come-First-Serve (FCFS)

B. Round-Robin

C. Shortest-Job-First (SJF)

D. Binary Search Tree (BST)

Hint

Separate class assignment, movement between queues and the rule for choosing work.

Answer D: Binary Search Tree is a data structure.

OS19 explicitly states that a binary search tree may implement a queue but is not itself a CPU selection policy. FCFS, Round Robin and SJF are actual scheduling algorithms described in OS15, OS18 and OS16. Consequently BST is the exclusion. Using a tree internally to maintain priorities would not change the selection policy into an algorithm called BST.

Why each choice is right or wrong

A - Incorrect: FCFS is a scheduling policy that runs ready bursts in their arrival order.

B - Incorrect: Round Robin is a scheduling policy assigning cyclic ready-queue turns bounded by a quantum.

C - Incorrect: SJF is a scheduling policy selecting the shortest next burst among ready jobs.

D - Correct: BST is a data structure that can support queue operations rather than a CPU scheduling policy.

Primary note proof

Study-note section OS19

> A binary search tree is a data structure that might implement a queue; it is not itself a CPU selection policy.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Question sources

Operating Systems practice questions · Question 4

Question 46 | Q01287

Which of the following is not a type of file system?

A. FAT32

B. NTFS

C. EXT4

D. HTTP

Hint

Distinguish a file operation, a resource handle, an offset and file metadata.

Answer D: HTTP is an application communication protocol.

OS36 explicitly names FAT32, NTFS and ext4 as filesystem formats and contrasts HTTP as an application communication protocol. Filesystems organize stored file content and metadata; a communication protocol does not become a filesystem merely because it transfers file contents. Therefore HTTP is the single option outside the stated filesystem-format category.

Why each choice is right or wrong

A - Incorrect: FAT32 is explicitly identified as a filesystem format, making it a member of the requested category.

B - Incorrect: NTFS is explicitly identified as a filesystem format, not a communication-protocol exclusion.

C - Incorrect: ext4 is explicitly identified as a filesystem format in the complete study note.

D - Correct: HTTP is an application communication protocol rather than a filesystem format for organizing stored files.

Primary note proof

Study-note section OS36

> Sockets also support ordinary read/write, alongside socket/connect/send operations, so read is not uniquely “not a network call.” FAT32, NTFS and ext4 are filesystem formats; HTTP is an application communication protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 48 | lines 5-8

> 13.1 File
> A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the file’s creator and user.

Question sources

Operating Systems practice questions · Question 5

Question 47 | Q01288

What is the purpose of a device driver in an operating system?

A. To manage user accounts

B. To manage hardware resources

C. To manage software applications

D. To manage network connections

Hint

Separate device hardware, device-specific software and general event or transfer mechanisms.

Answer B: To manage hardware resources

A driver implements I/O behavior for a device class and hides hardware peculiarities (PDF pp10,47). B is the closest option when hardware resources means its particular device; it should not imply one driver manages every resource.

Why each choice is right or wrong

A - Incorrect: Account administration is not the device-specific I/O job.

B - Correct: Device hardware I/O management is supported (pp10,47).

C - Incorrect: Applications use driver services but the driver does not manage applications generally.

D - Incorrect: Only a network-device driver concerns a network adapter; that is not every driver's purpose.

Primary note proof

Study-note section OS34

> A controller is device hardware exposing status, control and data registers. A driver converts OS requests into device-specific operations. Device independence and uniform naming hide brand-specific details; error handling reports/retries supported failures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 30-37

> I/O subsystem comprised of I/O devices and their corresponding driver software. Drivers hides the peculiarities of specific hardware devices from the user as the device driver knows the peculiarities of the specific device.
> Operating System manages the communication between user and device drivers.
> The major activities of an operating system with respect to I/O Operation.
> • I/O operation means read or write operation with any file or any specific I/O device.
> • Program may require any I/O device while running.
> • Operating system provides the access to the required I/O device when required.

Operating System module | PDF page 47 | lines 19-26

> 12.5 Device driver
> Device driver is a program or routine developed for an I/O device. A device driver implements I/O operations or behaviors on a specific class of devices. For example a system supports one or a number of multiple brands of terminals, all slightly different terminals may have a single terminal driver. In the layered structure of I/O system, device driver lies between interrupt handler and device independent I/O software. The job of a device driver is following.
> To accept request from the device independent software above it.
> To see to it that the request is executed.

Question sources

Operating Systems practice questions · Question 6

Question 48 | Q01289

Which listed forms can serve as user interfaces to an operating system? Treat possible interface forms generally, rather than features of a particular product. Select all that apply.

A. Command-line interface (CLI)

B. Graphical user interface (GUI)

C. Natural language interface (NLI)

D. Virtual reality interface (VRI)

Hint

Distinguish identity checks, permission checks and the particular security property at issue.

Answer A, B, C, D: Command-line interface (CLI); Graphical user interface (GUI); Natural language interface (NLI); Virtual reality interface (VRI)

Command lines, graphical interfaces, speech or natural-language interaction, and virtual-reality interaction are possible interface forms. An OS need not implement all of them for each form to count as a possible interface. Because the stated question asks about capability in general rather than a particular product's installed features, all four listed forms satisfy it.

Why each choice is right or wrong

A - Correct: A CLI accepts text commands as a user interface.

B - Correct: A GUI provides graphical controls for user interaction.

C - Correct: Natural-language or speech interaction is a possible interface form.

D - Correct: VR interaction is also a possible interface form.

Primary note proof

Study-note section OS39

> CLI, GUI, speech/natural language and VR are possible interfaces, not a closed list excluding the last two.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 14 | lines 18-23

> Interactivity refers that a User is capable to interact with computer system. Operating system does the following activities related to interactivity.
> • OS provides user an interface to interact with system.
> • OS managers input devices to take inputs from the user. For example, keyboard.
> • OS manages output devices to show outputs to the user. For example, Monitor.
> • OS Response time needs to be short since the user submits and waits for the result.

Operating System module | PDF page 54 | lines 18-20

commands of the operating system. It can be used to do various types of operations, call application programs etc.
Security - Linux provides user security using authentication features like password

Question sources

Operating Systems practice questions · Question 7

Question 49 | Q01290

Which of the following is not a component of an operating system kernel?

A. Process scheduler

B. Memory manager

C. File system

D. Web browser

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer D: Web browser.

OS03 defines the kernel through core resource management, protection and system-call handling, while explicitly calling browsers applications. Scheduling, memory management and filesystem service are core resource-management functions in the textbook kernel sense. Some systems move services outside a kernel, but a web browser remains the clear application-level exclusion from the listed ordinary kernel responsibilities.

Why each choice is right or wrong

A - Incorrect: Process scheduling is a core resource-management function in the textbook kernel model used here.

B - Incorrect: Memory management is a core resource-management function rather than the listed user application.

C - Incorrect: Filesystem service belongs to ordinary OS resource management in this textbook kernel-level comparison.

D - Correct: The note explicitly identifies browsers as applications rather than components implementing the kernel's core role.

Primary note proof

Study-note section OS03

The kernel implements core resource management, protection and system-call handling. A shell reads/interprets user commands; a GUI is another interface. Browsers, office tools and editors are applications.

Study-note section OS01

An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 11-15

operating system. It is consisting of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
System Library - System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries

Operating System module | PDF page 6 | lines 4-9

In multiprogramming environment, OS decides which process gets the processor when and how much time. This function is called process scheduling. Operating System does the following activities for processor management. Keeps tracks of processor and status of process. Program responsible for this task is known as traffic controller.
Allocates the processor (CPU) to a process.
De-allocates processor when processor is no longer required.

Operating System module | PDF page 31 | lines 24-31

Memory management is the functionality of an operating system which handles or manages primary memory. Memory management keeps track of each and every memory location either it is allocated to some process or it is free. It checks how much memory is to be allocated to processes. It decides which process will get memory at what time. It tracks whenever some memory gets freed or unallocated and correspondingly it updates the status.
Memory management provides protection by using two registers, a base register and a limit register. The base register holds the smallest legal physical memory address and the limit register specifies the size of the range. For example, if the base register holds 300000 and the

Question sources

Operating Systems practice questions · Question 8

Question 50 | Q01291

Which of the following is not a type of system call used in operating systems?

A. Process control

B. File management

C. Network communication

D. Video editing

Hint

Identify the abstraction layer and separate a protected service request from an application task.

Answer D: Video editing is an application operation.

OS04 lists process control, file/device management, information retrieval and communication as service categories. It explicitly distinguishes video editing as an application operation. Network communication falls within communication services, so it is not excluded merely because it uses sockets rather than local files. The category outside generic system calls is video editing.

Why each choice is right or wrong

A - Incorrect: Process control is explicitly included among protected operating-system service categories.

B - Incorrect: File management is explicitly included among operating-system services requested through system calls.

C - Incorrect: Network communication falls within the communication service category rather than being excluded from system calls.

D - Correct: Video editing is an application-level operation, not a standard generic system-call service category.

Primary note proof

Study-note section OS04

> Process control, file/device management, information retrieval and communication are service categories. Video editing and a database-specific write_database() are application operations, not standard generic syscall categories/names.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 5-15

> An Operating System provides services to both the users and to the programs.
> • It provides programs, an environment to execute.
> • It provides users, services to execute the programs in a convenient manner.
> Operating Systems can provide the following common services:
> ✓ Program execution
> ✓ I/O operations
> ✓ File System manipulation
> ✓ Communication
> ✓ Error Detection
> ✓ Resource Allocation
> ✓ Protection

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.

- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

Operating System module | PDF page 11 | lines 3-8

- Program needs to read a file or write a file.
- The operating system gives the permission to the program for operation on file.
- Permission varies from read-only, read-write, denied and so on.
- Operating System provides an interface to the user to create/delete files.
- Operating System provides an interface to the user to create/delete directories.
- Operating System provides an interface to create the backup of file system.

Operating System module | PDF page 11 | lines 15-19

- Two processes often require data to be transferred between them.
- The both processes can be on the one computer or on different computer but are connected through computer network.
- Communication may be implemented by two methods either by Shared Memory or by Message Passing.

Question sources

Operating Systems practice questions · Question 9

Question 51 | Q01292

What is the purpose of a shell in an operating system?

A. To manage user accounts

B. To manage hardware resources

C. To manage software applications

D. To provide a user interface

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer D: To provide a user interface

PDF p54 calls the shell an interface to the kernel that takes user commands. D captures its interface role rather than an individual resource-management service.

Why each choice is right or wrong

A - Incorrect: Account management may be requested through shell commands, but is not its defining role.

B - Incorrect: Kernel provides low-level hardware services (p54).

C - Incorrect: Invoking applications is supported, but does not replace the general interface role.

D - Correct: Directly follows the user-command interface description (p54).

Primary note proof

Study-note section OS03

> The kernel implements core resource management, protection and system-call handling. A shell reads/interprets user commands; a GUI is another interface. Browsers, office tools and editors are applications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 54 | lines 18-20

> commands of the operating system. It can be used to do various types of operations, call application programs etc.
> Security - Linux provides user security using authentication features like password

Operating System module | PDF page 54 | lines 24-30

> Hardware layer - Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc.).
> Kernel - Core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.
> Shell - An interface to kernel, hiding complexity of kernel's functions from users. Takes commands from user and executes kernel's functions.
> Utilities - Utility programs giving user most of the functionalities of an operating systems.

Question sources

Operating Systems practice questions · Question 10

Question 52 | Q01293

Which listed systems are proprietary OS products rather than open-source systems as a whole? Interpret Linux as the open-source kernel and Android as AOSP without vendor additions. Select all that apply.

A. Linux

B. Android

C. macOS

D. Windows

Hint

Use the stated historical or platform scope and distinguish origin from technical capability.

Answer C, D: macOS; Windows

Windows and macOS are proprietary products. The existence of open-source components within macOS does not make the complete product open source. Linux is treated here as the open-source kernel, and Android is explicitly restricted to AOSP, excluding proprietary applications or services bundled by device vendors. With that component-versus-product scope fixed, both macOS and Windows belong in the selected set.

Why each choice is right or wrong

A - Incorrect: Linux is the open-source kernel under the stated scope.

B - Incorrect: AOSP is open source; vendor additions are explicitly excluded.

C - Correct: macOS is a proprietary product even though some components are open source.

D - Correct: Windows is a proprietary OS product.

Primary note proof

Study-note section OS40

> Linux is a Unix-compatible kernel written from scratch by Linus Torvalds; a distribution combines it with userland, libraries and tools. Android's base kernel layer is Linux; its framework and applications are higher layers. AOSP is open source, while a shipped device can include proprietary applications/services. Windows and macOS are both proprietary products, although macOS contains open-source components, so "which is not open source?" offering both is ambiguous. Microsoft Office is an application suite.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 5-7

> Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

Operating System module | PDF page 54 | lines 5-17

> Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
> Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
> Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
> Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
> Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
> Shell - Linux provides a special interpreter program which can be used to execute

Question sources

Operating Systems practice questions · Question 11

Question 53 | Q01294

What is the purpose of a bootloader in an operating system?

A. To load the operating system kernel into memory

B. To manage hardware resources

C. To manage software applications

D. To provide a user interface

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer A: Load the OS kernel into memory.

OS03 places firmware/bootloader stages before transfer of control to a kernel loaded into RAM. This identifies loading the kernel as the bootloader purpose among the alternatives. Resource management is subsequently a kernel role, application management is a running-system function, and user interfaces are provided by shells/GUI components rather than defining the bootloader's purpose.

Why each choice is right or wrong

A - Correct: The bootloader prepares the kernel in memory and transfers control to it during the boot sequence.

B - Incorrect: Core hardware-resource management is the kernel's role after the bootloader hands over control.

C - Incorrect: Managing running applications belongs to the operating system's runtime services rather than defining the bootloader.

D - Incorrect: Shells and GUIs provide user interfaces; that is not the bootloader's principal role here.

Primary note proof

Study-note section OS03

> Firmware/bootloader stages execute before transferring control to a kernel loaded into RAM.

Study-note section OS03

> The kernel implements core resource management, protection and system-call handling. A shell reads/interprets user commands; a GUI is another interface. Browsers, office tools and editors are applications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 11-15

> operating system. It is consisting of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
> System Library - System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries

Operating System module | PDF page 32 | lines 11-14

> In dynamic loading, a routine of a program is not loaded until it is called by the program. All routines are kept on disk in a re-locatable load format. The main program is loaded into memory and is executed. Other routines methods or modules are loaded on request. Dynamic loading makes better memory space utilization and unused routines are never loaded.

Question sources

Operating Systems practice questions · Question 12

Question 54 | Q01295

What is the difference between a kernel-level thread and a user-level thread?

A. Kernel-level threads are faster than user-level threads

B. Kernel-level threads are managed by the operating system, while user-level threads are managed by the application

C. Kernel-level threads can access hardware resources directly, while user-level threads cannot

D. There is no difference between a kernel-level thread and a user-level thread

Hint

Compare per-thread execution context with shared process resources and the stated thread mapping.

Answer B: Kernel-level threads are managed by the operating system, while user-level threads are managed by the application

Application management of user threads is explicit on PDF p28; kernel management is explicit on p29. Kernel threads are generally slower to create/manage, and management location does not automatically grant application thread code kernel privilege.

Why each choice is right or wrong

A - Incorrect: Opposes the p29 slower-to-create/manage comparison.

B - Correct: Exactly matches management location on pp28-29.

C - Incorrect: Confuses scheduling/management by the kernel with execution privilege; p53 distinguishes user code from kernel code.

D - Incorrect: Management location and costs explicitly differ.

Primary note proof

Study-note section OS08

User-level threads are managed by a library; kernel threads are known to the OS. In many-to-one mapping, a blocking kernel call can block all user threads and one kernel execution context cannot use multiple CPUs simultaneously. One-to-one permits another kernel thread to run when one blocks. Many-to-many multiplexes user threads over several kernel threads. Speed and reliability depend on implementation; the management layer alone guarantees neither. Application names such as browser or Excel do not prove a particular version's threading model.

Study-note section OS-S02

In the conventional library-thread comparison, user-level thread creation and switching can avoid entering the kernel. Kernel-managed thread creation and management commonly incur additional kernel work and are therefore generally slower to create and manage in this comparison. This is a typical comparison, not a universal speed or reliability guarantee.

Study-note section OS-S02

Being scheduled or managed by the kernel does not give application thread code kernel privilege. Ordinary applications and supporting utilities execute in restricted user mode and request protected services through libraries and system calls. Kernel component code executes in privileged kernel mode and performs permitted low-level hardware operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 28 | lines 20-29

In this case, application manages thread management kernel is not aware of the existence of threads. The thread library contains code for creating and destroying threads, for passing message and data between threads, for scheduling thread execution and for saving and restoring thread contexts. The application begins with a single thread and begins running in that thread.
Advantages
• Thread switching does not require Kernel mode privileges.
• User level thread can run on any operating system.
• Scheduling can be application specific in the user level thread.
• User level threads are fast to create and manage.

Operating System module | PDF page 29 | lines 4-16

In this case, thread management done by the Kernel. There is no thread management code in the application area. Kernel threads are supported directly by the operating system. Any application can be programmed to be multithreaded. All of the threads within an application are supported within a single process.
The Kernel maintains context information for the process as a whole and for individuals' threads within the process. Scheduling by the Kernel is done on a thread basis. The Kernel performs thread creation, scheduling and management in Kernel space. Kernel threads are

generally slower to create and manage than the user threads.
Advantages
• Kernel can simultaneously schedule multiple threads from the same process on multiple processes.
• If one thread in a process is blocked, the Kernel can schedule another thread of the same process.

Operating System module | PDF page 53 | lines 21-29

Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast. Kernel runs each process and provides system services to processes, provides protected access to hardware to processes.
Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

Question sources

Operating Systems practice questions · Question 13

Question 55 | Q01296

Which listed item is a general data structure rather than a deadlock-specific avoidance method or dependency-graph analysis?

A. Banker's algorithm

B. Wait-for graph algorithm

C. Resource allocation graph algorithm

D. Binary search tree algorithm

Hint

Compute remaining needs and simulate feasible completions before deciding whether an allocation is safe.

Answer D: Binary search tree algorithm

Banker’s algorithm checks whether tentative allocations leave a safe completion sequence and is an avoidance method. Wait-for and resource-allocation graphs represent dependencies used in deadlock reasoning. A binary search tree is a general data structure and does not, by its name alone, specify deadlock handling. The broader category distinguishes the unrelated item without misclassifying avoidance or detection as strict prevention.

Why each choice is right or wrong

A - Incorrect: Banker is a deadlock-avoidance method, so it belongs to the stated handling category.

B - Incorrect: A wait-for graph directly represents process dependencies for deadlock analysis.

C - Incorrect: A resource-allocation graph represents resource requests and assignments for deadlock reasoning.

D - Correct: A binary search tree is a general search structure rather than a deadlock-specific method.

Primary note proof

Study-note section OS25

> Avoidance allows requests only if the resulting state remains safe. Banker requires maximum claims. Compute Need=Max−Allocation.

Study-note section OS23

> In a resource-allocation graph, P→R is a request and an instance R→P is an assignment. Follow these directions to form waits. No cycle implies no deadlock; with one instance of every resource type, a cycle implies deadlock. With multiple instances a cycle alone is insufficient: another instance may permit completion. A wait-for graph replaces single-instance resource nodes with P→Q when P waits for Q.

Study-note section OS19

> A binary search tree is a data structure that might implement a queue; it is not itself a CPU selection policy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Operating Systems practice questions · Question 14

Question 56 | Q01297

What is the general execution role of a process in an operating system, regardless of the application it runs?

A. To manage user accounts

B. To manage hardware resources

C. To execute a program instance with its own address space and resources

D. To provide a user interface

Hint

Distinguish stored instructions, an executing instance and its saved execution state.

Answer C: To execute a program instance with its own address space and resources

A process is an executing instance of a program, accompanied by an address space and resources. Its application may administer accounts, interact with hardware through services, or display a user interface, but none of those specialized jobs defines every process. The execution option expresses the general role. It also distinguishes a running instance from a program stored as instructions.

Why each choice is right or wrong

A - Incorrect: Account administration is only one possible application responsibility.

B - Incorrect: Hardware management is an OS responsibility rather than the general role of every process.

C - Correct: Executing a program instance with an address space and resources is the process abstraction.

D - Incorrect: Many processes run without providing a user interface.

Primary note proof

Study-note section OS05

> A program is stored instructions; a process is an executing instance with address space and resources. A process does not universally “manage applications” or “provide a user interface.” The PCB records identity, state, PC/registers, scheduling information, memory/accounting details and I/O/open-resource state.

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 6-9

A process is a program in execution. The execution of a process must progress in a sequential fashion. Definition of process is following.
• A process is defined as an entity which represents the basic unit of work to be implemented in the system.

Operating System module | PDF page 5 | lines 6-13

An operating System (OS) is an intermediary between users and computer hardware. It provides users an environment in which a user can execute programs conveniently and efficiently. In technical terms, it is software which manages hardware. An operating System controls the allocation of resources and services such as memory, processors, devices and information.
1.1 Definition
An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Question sources

Operating Systems practice questions · Question 15

Question 57 | Q01298

Which of the following is not a type of process state in an operating system?

A. Ready

B. Running

C. Blocked

D. Sleeping Beauty

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer D: Sleeping Beauty

The state list on PDF p16 contains Ready, Running and Waiting. Interpret Blocked as the waiting-for-event state. Sleeping Beauty is not one of this module's five named states.

Why each choice is right or wrong

A - Incorrect: Ready is explicitly listed.

B - Incorrect: Running is explicitly listed.

C - Incorrect: Blocked is the conventional paraphrase of waiting for an event; Waiting is the module's exact term.

D - Correct: Not present in the five-state model, and not a defined state term.

Primary note proof

Study-note section OS06

New processes are created/admitted; ready means eligible for CPU but not currently running. Running uses a CPU. Waiting/blocked means unable to proceed until an event; terminated means execution has ended. Running→ready follows time-slice expiry or another preemption. Running→waiting follows blocking I/O. I/O completion normally gives waiting→ready, not a guaranteed immediate running state.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 29-39

1. New The process is being created.
2. Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
3. Running Process instructions are being executed (i.e. The process that is currently being executed).
4. Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
5. Terminated The process has finished execution.

Question sources

Operating Systems practice questions · Question 16

Question 58 | Q01299

What is the difference between a process and a thread in an operating system?

A. A process is a unit of execution, while a thread is a unit of scheduling

B. A process is a unit of scheduling, while a thread is a unit of execution

C. There is no difference between a process and a thread

D. A process can have multiple threads, while a thread can only belong to one process

Hint

Compare per-thread execution context with shared process resources and the stated thread mapping.

Answer D: A process can have multiple threads, while a thread can only belong to one process

PDF p27 says every thread belongs to exactly one process and separate flows improve parallel execution; p29 supports multiple threads within one process. D directly expresses that relationship.

Why each choice is right or wrong

A - Incorrect: Processes and threads can both be scheduled (pp20,29); this is not an exclusive distinction.

B - Incorrect: Both may participate in scheduling and execution; reversal does not define their containment.

C - Incorrect: PDF p27 explicitly distinguishes heavy processes from light threads.

D - Correct: Directly follows the one-process-per-thread and multithreaded-process definitions (pp27,29).

Primary note proof

Study-note section OS08

> A thread has its own PC, registers and stack. Threads in a process share code, address space and usually open-file resources. A process can contain many threads; a thread belongs to one process at a time. Threads are not a group of processes and need not execute sequentially. Independent processes can exist; cooperating ones communicate/share data.

Study-note section OS08

> User-level threads are managed by a library; kernel threads are known to the OS. In many-to-one mapping, a blocking kernel call can block all user threads and one kernel execution context cannot use multiple CPUs simultaneously. One-to-one permits another kernel thread to run when one blocks. Many-to-many multiplexes user threads over several kernel threads.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 27 | lines 6-14

> A thread is a flow of execution through the process code, with its own program counter, system registers and stack. A thread is also called a light weight process. Threads provide a way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.
> Each thread belongs to exactly one process and no thread can exist outside a process. Each thread represents a separate flow of control. Threads have been successfully used in implementing network servers and web server. They also provide a suitable foundation for parallel execution of applications on shared memory multiprocessors. Following figure shows

Operating System module | PDF page 29 | lines 4-16

> In this case, thread management done by the Kernel. There is no thread management code in the application area. Kernel threads are supported directly by the operating system. Any application can be programmed to be multithreaded. All of the threads within an application are supported within a single process.
> The Kernel maintains context information for the process as a whole and for individuals' threads within the process. Scheduling by the Kernel is done on a thread basis. The Kernel performs thread creation, scheduling and management in Kernel space. Kernel threads are generally slower to create and manage than the user threads.

Advantages
• Kernel can simultaneously schedule multiple threads from the same process on multiple processes.
• If one thread in a process is blocked, the Kernel can schedule another thread of the same process.

Operating System module | PDF page 20 | lines 15-21

It is also called CPU scheduler. Main objective is increasing system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects process among the processes that are ready to execute and allocates CPU to one of them.
Short term scheduler also known as dispatcher, execute most frequently and makes the fine-grained decision of which process to execute next. Short term scheduler is faster than long term scheduler.

Question sources

Operating Systems practice questions · Question 17

Question 59 | Q01300

Which of the following is not a type of synchronization mechanism used in operating systems?

A. Mutex

B. Semaphore

C. Monitor

D. Keyboard

Hint

Track available permits and blocked operations using the stated semaphore model.

Answer D: Keyboard.

OS10 describes monitors as protected operations with condition synchronization, and OS11 explains mutex/semaphore exclusion mechanisms. These coordinate concurrent execution. A keyboard is an input device rather than such a coordination abstraction. Interrupts generated by hardware may participate in event handling, but that does not make the keyboard itself a mutex, semaphore or monitor mechanism.

Why each choice is right or wrong

A - Incorrect: A mutex provides owned mutual exclusion and is a synchronization mechanism in the note.

B - Incorrect: A semaphore coordinates participants through atomic wait and signal operations, so it is a synchronization mechanism.

C - Incorrect: A monitor combines protected operations and condition synchronization, making it a genuine synchronization abstraction.

D - Correct: A keyboard is a physical input device rather than a process/thread synchronization abstraction.

Primary note proof

Study-note section OS11

> A semaphore provides atomic wait/P/down and signal/V/up operations. Under the nonnegative-token model, wait consumes one available token or blocks when none is available; signal supplies a token and may wake a waiter. Implementations using negative internal counts encode waiters differently, so state the model. A binary semaphore represents one permit; a counting semaphore tracks several. A mutex normally has ownership rules that a general semaphore lacks.

Study-note section OS10

> A barrier waits until a defined set of participants reaches a point; a monitor encapsulates protected operations with condition synchronization.

Study-note section OS34

> Input/output/storage/NIC devices are OS-managed hardware; user accounts are identities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Operating System module | PDF page 43 | lines 7-10

> Computers operate on many kinds of devices. General types include storage devices (disks, tapes), transmission devices (network cards, modems), and human-interface devices (screen, keyboard, mouse). Other devices are more specialized. A device communicates with a computer system by sending signals over a cable or even through the air.

Question sources

Operating Systems practice questions · Question 18

Question 60 | Q01301

What is the purpose of a context switch in an operating system?

A. To save the state of a process or thread and switch to another process or thread

B. To manage hardware resources

C. To manage software applications

D. To provide a user interface

Hint

Distinguish stored instructions, an executing instance and its saved execution state.

Answer A: To save the state of a process or thread and switch to another process or thread

PDF p21 defines saving/restoring CPU context so execution can resume, enabling multiple processes to share a CPU. A identifies that specific operation.

Why each choice is right or wrong

A - Correct: Directly states save state and switch execution (p21).

B - Incorrect: Resource management is the wider OS purpose, not the specific state operation.

C - Incorrect: Application management is broader and omits saved execution state.

D - Incorrect: A user interface is not what a context switch provides.

Primary note proof

Study-note section OS05

> A context switch saves one execution context and restores another; useful application work is interrupted by this overhead.

Study-note section OS08

> A thread has its own PC, registers and stack.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 21 | lines 41-50

> A context switch is the mechanism to store and restore the state or context of a CPU in Process Control block so that a process execution can be resumed from the same point at a later time. Using this technique, a context switcher enables multiple processes to share a single CPU. Context switching is an essential part of a multitasking operating system features.
> When the scheduler switches the CPU from executing one process to execute another, the context switcher saves the content of all processor registers for the process being removed from the CPU, in its process descriptor. The context of a process is represented in the

process control block of a process. Context switch time is pure overhead. Context switching can significantly affect performance as modern computers have a lot of general

Question sources

Operating Systems practice questions · Question 19

Question 61 | Q01302

Which item is an application-layer protocol rather than a primitive OS IPC mechanism such as a stream, shared region or message queue?

A. Pipes

B. Shared memory

C. Message queues

D. HTTP requests

Hint

Identify how data moves, which participant waits and what event allows the operation to continue.

Answer D: HTTP requests

Pipes provide streams, shared memory provides a common region, and message queues preserve messages. These are primitive IPC mechanisms in the stated classification. HTTP is an application protocol that can use an underlying communication mechanism such as sockets. It can certainly carry interprocess communication, but it is the only listed application-layer protocol rather than one of the primitive mechanisms requested here.

Why each choice is right or wrong

A - Incorrect: Pipes are primitive stream-based IPC mechanisms.

B - Incorrect: Shared memory is a primitive common-region IPC mechanism.

C - Incorrect: Message queues are primitive message-preserving IPC mechanisms.

D - Correct: HTTP requests use an application-layer protocol over an underlying mechanism.

Primary note proof

Study-note section OS09

Shared memory lets processes access a common region and requires synchronization for conflicting updates. Message passing exchanges data through send/receive operations. Pipes provide streams; message queues preserve messages; sockets connect endpoints; RPC requests a remote operation. HTTP is an application protocol over a communication mechanism and can carry interprocess communication, so it is not universally “not IPC.”

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 11 | lines 15-19

> • Two processes often require data to be transferred between them.
> • The both processes can be on the one computer or on different computer but are connected through computer network.
> • Communication may be implemented by two methods either by Shared Memory or by Message Passing.

Question sources

Operating Systems practice questions · Question 20

Question 62 | Q01303

What is the general purpose of a system call in an operating system?

A. To manage user accounts

B. To request a protected kernel service

C. To manage software applications

D. To provide a user interface

Hint

Identify the abstraction layer and separate a protected service request from an application task.

Answer B: To request a protected kernel service

A system call is the mechanism through which executing code requests a protected kernel service. Process creation, file reads and socket operations are examples. A program that manages accounts, applications or a user interface may use system calls, but those tasks are not the mechanism’s general purpose. The option captures the common role without restricting it to one kind of resource.

Why each choice is right or wrong

A - Incorrect: Account management can use system calls but does not define their general purpose.

B - Correct: Requesting a protected kernel service states the general purpose.

C - Incorrect: Managing applications is a possible use rather than the system-call mechanism’s definition.

D - Incorrect: A UI may issue calls, but providing a UI is not the general system-call role.

Primary note proof

Study-note section OS04

A system call requests a protected kernel service, such as creating a process, reading a file or opening a socket. A user-facing API/library may wrap that request; not every library function enters the kernel.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 16-19

implement most of the functionalities of the operating system and do not requires kernel module's code access rights.
System Utility - System Utility programs are responsible to do specialized, individual level tasks.

Operating System module | PDF page 45 | lines 28-33

Application I/O Interface represents the structuring techniques and interfaces for the operating system to enable I/O devices to be treated in a standard, uniform way. The actual differences lies kernel level modules called device drivers which are custom tailored to corresponding devices but show one of the standard interfaces to applications. The purpose of the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, such as the I/O system calls. Following are the characteristics of I/O

Question sources

Operating Systems practice questions · Question 21

Question 63 | Q01305

Which of the following is not a type of scheduling policy used in operating systems?

A. Preemptive

B. Non-preemptive

C. Cooperative

D. Infinite

Hint

Check whether the running burst can be interrupted before completion or blocking.

Answer D: Infinite is not a scheduling-policy category.

OS14 distinguishes preemptive and non-preemptive scheduling and explains cooperative scheduling as relying on voluntary release. OS19 explicitly rejects Infinite as a standard scheduling-policy category. Thus the first three labels describe genuine scheduling approaches, whereas infinite does not identify a selection or preemption rule. It should not be confused with a possible unbounded waiting-time outcome.

Why each choice is right or wrong

A - Incorrect: Preemptive scheduling can interrupt running work and return it to the ready state.

B - Incorrect: Non-preemptive scheduling retains a running burst until completion, blocking or voluntary release.

C - Incorrect: Cooperative scheduling is the recognized approach relying on voluntary release of execution.

D - Correct: Infinite names no scheduling selection or preemption policy in the provided note.

Primary note proof

Study-note section OS14

> Non-preemptive scheduling keeps a running CPU burst until completion, blocking or voluntary yielding. Preemptive scheduling can return running work to ready, usually after a timer quantum or a more urgent arrival. FCFS and non-preemptive SJF are non-preemptive; RR and SRTF are preemptive. Priority can be either, so its name alone cannot decide. Cooperative describes relying on voluntary release.

Study-note section OS19

> “Infinite” is not a standard scheduling-policy category.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.
> • Easy to understand and implement.
> • Poor in performance as average wait time is high.
> Wait time of each process is following
> Process Wait Time: Service Time - Arrival Time

Question sources

Operating Systems practice questions · Question 23

Question 64 | Q01306

24.What is the purpose of a page table in memory management?

A. To translate virtual addresses to physical addresses

B. To manage user accounts

C. To manage software applications

D. To provide a user interface

Hint

Split the address into page and offset, then apply the mapping and access checks.

Answer A: To translate virtual addresses to physical addresses

Page number indexes a table containing physical page bases (PDF p34); combining the base with the page offset yields the physical address (p35). Thus a page table participates in virtual-to-physical translation, A.

Why each choice is right or wrong

A - Correct: Matches page-table address lookup (pp34-35).

B - Incorrect: User accounts are not the page-number-to-frame mapping.

C - Incorrect: Application management is broader than the translation data structure.

D - Incorrect: An execution address map does not provide the user interface.

Primary note proof

Study-note section OS30

> Pages and physical frames have equal fixed size P. Divide virtual address v into page=floor(v/P) and offset=v mod P. The page table maps a page to a frame and protection/status flags. Physical address=frame×P+offset. A TLB caches translations; a TLB miss may require a table walk while the page is already resident.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 34 | lines 47-56

> External fragmentation is avoided by using paging technique. Paging is a technique in which physical memory is broken into blocks of the same size called pages (size is power of 2, between 512 bytes and 8192 bytes). When a process is to be executed, it's corresponding pages are loaded into any available memory frames.
> Logical address space of a process can be non-contiguous and a process is allocated physical memory whenever the free memory frame is available. Operating system keeps track of all free frames. Operating system needs n free frames to run a program of size n pages.
> Address generated by CPU is divided into
> • Page number (p) -- page number is used as an index into a page table which contains base address of each page in physical memory.

Operating System module | PDF page 33 | lines 6-13

> The run-time mapping from virtual to physical address is done by the memory management unit (MMU) which is a hardware device. MMU uses following mechanism to convert virtual address to physical address.
> • The value in the base register is added to every address generated by a user process which is treated as offset at the time it is sent to memory. For example, if the base register value is 10000, then an attempt by the user to use address location 100 will be dynamically reallocated to location 10100.
> • The user program deals with virtual addresses; it never sees the real physical addresses.

Operating System module | PDF page 35 | lines 3-4

> • Page offset (d) -- page offset is combined with base address to define the physical memory address.

Question sources

Operating Systems practice questions · Question 24

Question 65 | Q01308

What is the purpose of a system monitor in an operating system?

A. To manage user accounts

B. To manage hardware resources

C. To manage software applications

D. To provide information about system performance and resource usage

Hint

Identify whether data is in transit, retained for reuse or queued for a device.

Answer D: Report performance and resource usage.

OS35 says a system monitor reports CPU, memory, storage and network usage and measures rather than creating more resources. This matches providing information about performance and usage. User-account administration, hardware allocation and application management are broader OS/administration functions, not the monitor's defining reporting role. Observing a resource is distinct from allocating or increasing it.

Why each choice is right or wrong

A - Incorrect: Account management is an administration function, not the monitor's specified performance-reporting purpose.

B - Incorrect: The monitor measures hardware usage rather than serving as the general resource-allocation manager.

C - Incorrect: Application management is broader than the monitor's role of reporting system performance and consumption.

D - Correct: Reporting system performance and resource usage directly matches the monitor definition in the note.

Primary note proof

Study-note section OS35

> A system monitor reports CPU, memory, storage and network usage; it measures rather than directly creating more resources.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 31-35

> Control over system performance -- Recording delays between request for a service and response from the system.
> Job accounting -- Keeping track of time and resources used by various jobs and users.
> Error detecting aids -- Production of dumps, traces, error messages and other debugging and error detecting aids.

Question sources

Operating Systems practice questions · Question 26

Question 66 | Q01309

What is the difference between a kernel and a shell in an operating system?

A. A kernel is the core of the operating system, while a shell provides a user interface

B. A kernel provides a user interface, while a shell manages hardware resources

C. There is no difference between a kernel and a shell

D. A kernel manages hardware resources, while a shell manages software applications

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer A: A kernel is the core of the operating system, while a shell provides a user interface

The kernel is the core interacting with hardware (PDF p53); the shell takes user commands and interfaces to the kernel (p54). A preserves both roles.

Why each choice is right or wrong

A - Correct: Both halves match pp53-54.

B - Incorrect: Reverses the explicit roles.

C - Incorrect: The module lists and distinguishes the two layers.

D - Incorrect: First half is supported; invoking applications is only part of shell use, and omits its defining command interface.

Primary note proof

Study-note section OS03

> The kernel implements core resource management, protection and system-call handling. A shell reads/interprets user commands; a GUI is another interface. Browsers, office tools and editors are applications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 11-15

> operating system. It is consisting of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
> System Library - System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries

Operating System module | PDF page 54 | lines 18-20

> commands of the operating system. It can be used to do various types of operations, call application programs etc.
> Security - Linux provides user security using authentication features like password

Operating System module | PDF page 54 | lines 24-30

> Hardware layer - Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc.).
> Kernel - Core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.
> Shell - An interface to kernel, hiding complexity of kernel's functions from users. Takes commands from user and executes kernel's functions.
> Utilities - Utility programs giving user most of the functionalities of an operating systems.

Question sources

Operating Systems practice questions · Question 27

Question 67 | Q01310

Which of the following is not a type of system call used in file management?

A. open()

B. close()

C. read()

D. write_database()

Hint

Distinguish a file operation, a resource handle, an offset and file metadata.

Answer D: write_database() is not a standard generic file-management system call.

OS36 identifies open, close and read as ordinary file operations, and OS04 explicitly distinguishes database-specific write_database() as an application operation rather than a standard generic syscall name. A database operation may internally use actual file syscalls, but that composition does not make its application-specific name a generic kernel file-management call.

Why each choice is right or wrong

A - Incorrect: open obtains an open-file reference and is a standard file-management operation described here.

B - Incorrect: close releases an open-file reference and is a standard file-management operation.

C - Incorrect: read transfers bytes from an open resource and is a standard generic file operation.

D - Correct: write_database is an application-specific database operation rather than a standard generic file-management syscall.

Primary note proof

Study-note section OS36

> A file has content and metadata such as name, size, owner, permissions and timestamps. open obtains an open-file reference, read/write transfer bytes, seek changes an offset, close releases a reference, truncate changes size and unlink removes a name.

Study-note section OS04

> Process control, file/device management, information retrieval and communication are service categories. Video editing and a database-specific write_database() are application operations, not standard generic syscall categories/names.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 11 | lines 3-8

> • Program needs to read a file or write a file.
> • The operating system gives the permission to the program for operation on file.
> • Permission varies from read-only, read-write, denied and so on.
> • Operating System provides an interface to the user to create/delete files.
> • Operating System provides an interface to the user to create/delete directories.
> • Operating System provides an interface to create the backup of file system.

Question sources

Operating Systems practice questions · Question 28

Question 68 | Q01311

Which item is a hardware/CPU event-notification mechanism rather than a high-level lock, participant rendezvous, or protected-operation abstraction?

A. Locks

B. Barriers

C. Monitors

D. Interrupts

Hint

Identify conflicting shared accesses and the safety or progress property required.

Answer D: Interrupts

Locks coordinate entry to protected state, barriers wait for participants to reach a common point, and monitors encapsulate protected operations with conditions. Interrupts notify the CPU of events and may help coordinate work, but belong to a different abstraction level. The classification therefore selects interrupts without denying that interrupts can participate in synchronization or completion notification.

Why each choice is right or wrong

A - Incorrect: A lock is a high-level mutual-exclusion abstraction.

B - Incorrect: A barrier is a participant rendezvous abstraction.

C - Incorrect: A monitor encapsulates protected operations and condition synchronization.

D - Correct: An interrupt is a CPU event-notification mechanism, the different abstraction level requested.

Primary note proof

Study-note section OS10

> Keep critical regions small and ensure every conflicting access uses the same protection. A barrier waits until a defined set of participants reaches a point; a monitor encapsulates protected operations with condition synchronization.

Study-note section OS11

> A mutex normally has ownership rules that a general semaphore lacks.

Study-note section OS34

> Polling checks readiness repeatedly; interrupts request CPU attention; DMA transfers blocks after setup without CPU work for every byte.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

• Loads a program into memory.
• Executes the program.
• Handles program's execution.
• Provides a mechanism for process synchronization.
• Provides a mechanism for process communication.
• Provides a mechanism for deadlock handling.

Operating System module | PDF page 45 | lines 10-22

The CPU hardware uses an interrupt request line wire which helps CPU to sense after executing every instruction. When the CPU checks that a controller has put a signal on the interrupt request line, the CPU saves a state, such as the current value of the instruction pointer, and jumps to the interrupt handler routine at a fixed address. The interrupt handler part determines the cause of the interrupt performs the necessary processing and executes an interrupt instruction to return the CPU to its execution state.
The basic mechanism of interrupt enables the CPU to respond to an asynchronous event, such as when a device controller becomes ready for service. Most CPUs have two interrupt request lines.
• non-maskable interrupt - Such kind of interrupts are reserved for events like unrecoverable memory errors.
• maskable interrupt - Such interrupts can be switched off by the CPU before the execution of critical instructions that must not be interrupted.

Question sources

Operating Systems practice questions · Question 29

Question 69 | Q01312

What is the purpose of a virtual machine in an operating system?

A. To manage user accounts

B. To manage hardware resources

C. To manage software applications

D. To provide a virtualized environment for running multiple operating systems on a single physical machine

Hint

Identify the abstraction layer and separate a protected service request from an application task.

Answer D: A virtualized environment can run guest operating systems on one physical host.

OS04 defines virtualization as presenting a virtual machine environment, often allowing guest operating systems on one host through a hypervisor. That is exactly the complete description in D. Account, hardware and application management may occur within host or guest systems, but those broad administrative roles do not define why a virtual machine is provided.

Why each choice is right or wrong

A - Incorrect: User-account management is not the defining purpose of presenting a virtual machine environment.

B - Incorrect: Hardware is mediated, but general hardware management alone omits the guest-machine environment requested.

C - Incorrect: Application management does not define a virtual machine or distinguish it from an ordinary running OS.

D - Correct: Providing guest-machine environments on one physical host directly matches the note's virtualization definition.

Primary note proof

Study-note section OS04

> Virtualization presents a virtual machine environment, often allowing guest operating systems on one host through a hypervisor. It differs from virtual memory, which translates a process's addresses. Kernel, shell, API and VM are not interchangeable labels.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 37 | lines 6-11

> Virtual memory is a technique that allows the execution of processes which are not completely available in memory. The main visible advantage of this scheme is that programs can be larger than physical memory. Virtual memory is the separation of user logical memory from physical memory.
> This separation allows an extremely large virtual memory to be provided for programmers when only a smaller physical memory is available. Following are the situations, when entire

Question sources

Operating Systems practice questions · Question 30

Question 70 | Q01313

Which item coordinates access but does not itself transfer arbitrary data payloads or request an operation in another process?

A. Message passing

B. Shared memory

C. Remote procedure call (RPC)

D. File locking

Hint

Identify how data moves, which participant waits and what event allows the operation to continue.

Answer D: File locking

Message passing transfers messages, shared memory carries data through a common region, and RPC requests an operation in another process or machine. File locking instead coordinates who may access a file or region. Locks can therefore be part of IPC synchronization while not being a payload-transfer mechanism themselves. The narrower wording makes file locking the unique match without denying its coordination role.

Why each choice is right or wrong

A - Incorrect: Message passing transfers data using send/receive operations.

B - Incorrect: Shared memory carries data in a common accessible region.

C - Incorrect: RPC communicates a request to perform another process's operation.

D - Correct: File locking coordinates access rather than carrying arbitrary payloads itself.

Primary note proof

Study-note section OS09

> Shared memory lets processes access a common region and requires synchronization for conflicting updates. Message passing exchanges data through send/receive operations. Pipes provide streams; message queues preserve messages; sockets connect endpoints; RPC requests a remote operation.

Study-note section OS09

> File locks coordinate access rather than transfer arbitrary payloads, though locking is synchronization within IPC.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 11 | lines 15-19

> • Two processes often require data to be transferred between them.
> • The both processes can be on the one computer or on different computer but are connected through computer network.
> • Communication may be implemented by two methods either by Shared Memory or by Message Passing.

Question sources

Operating Systems practice questions · Question 31

Question 71 | Q01314

What is the purpose of a file system in an operating system?

A. To manage user accounts

B. To manage hardware resources

C. To manage software applications

D. To manage files and directories on storage devices

Hint

Distinguish a file operation, a resource handle, an offset and file metadata.

Answer D: To manage files and directories on storage devices

The file system tracks file information and locations and organizes directories (PDF p6); files store information on secondary storage (p48). D captures these functions.

Why each choice is right or wrong

A - Incorrect: Account administration manages user identities and accounts, rather than organizing stored file contents and directory names.

B - Incorrect: Hardware-resource management spans processors, memory and devices; it is broader than the file system's file-and-directory responsibility.

C - Incorrect: Application management does not describe file/directory organization.

D - Correct: Matches locations, directory organization and secondary storage (pp6,48).

Primary note proof

Study-note section OS36

> A file has content and metadata such as name, size, owner, permissions and timestamps. open obtains an open-file reference, read/write transfer bytes, seek changes an offset, close releases a reference, truncate changes size and unlink removes a name.

Study-note section OS37

> A single-level directory requires unique names in one namespace. Two-level directories separate users. A tree allows nested directories and absolute/relative paths; links can create shared graphs. Modern general-purpose systems commonly present hierarchical namespaces, but a feature name alone does not prove that every system excludes simpler arrangements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 19-23

> A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Operating System does the following activities for file management.

Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.

Operating System module | PDF page 48 | lines 5-8

13.1 File
A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the file's creator and user.

Question sources

Operating Systems practice questions · Question 32

Question 72 | Q01315

In this question only, "task" means a thread within a process. Which statement describes the process-task ownership relationship?

A. There is no difference between a process and a task

B. A process is a unit of execution, while a task is a unit of scheduling

C. A process can have multiple tasks, while a task can only belong to one process

D. A task is a unit of execution, while a process is a unit of scheduling

Hint

Compare per-thread execution context with shared process resources and the stated thread mapping.

Answer C: A process can have multiple tasks, while a task can only belong to one process

With task explicitly defined to mean thread, a process may contain several tasks while each task belongs to one process at a time. The tasks share the process address space but have their own execution context. This ownership relationship does not follow from the word task universally; it follows from the definition supplied here. Execution-versus-scheduling slogans do not state that relationship.

Why each choice is right or wrong

A - Incorrect: A thread/task is not identical to its containing process under this definition.

B - Incorrect: The execution-versus-scheduling description does not express the defined ownership relationship.

C - Correct: A process can contain multiple threads/tasks, each belonging to one process at a time.

D - Incorrect: The reversed execution/scheduling labels do not describe thread ownership.

Primary note proof

Study-note section OS08

> A thread has its own PC, registers and stack. Threads in a process share code, address space and usually open-file resources. A process can contain many threads; a thread belongs to one process at a time. Threads are not a group of processes and need not execute sequentially. Independent processes can exist; cooperating ones communicate/share data.

Study-note section OS05

> Task is OS-specific terminology: it may mean process, thread or schedulable entity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 6-9

> A process is a program in execution. The execution of a process must progress in a sequential fashion. Definition of process is following.
> • A process is defined as an entity which represents the basic unit of work to be implemented in the system.

Operating System module | PDF page 17 | lines 4-7

> Each process is represented in the operating system by a process control block (PCB) also called a task control block. PCB is the data structure used by the operating system. Operating system groups all information that needs about particular process. PCB contains many pieces of information associated with a specific process which is described below.

Question sources

Operating Systems practice questions · Question 33

Question 73 | Q01316

Which of the following is not a type of memory allocation algorithm used in operating systems?

A. First-Fit

B. Best-Fit

C. Next-Fit

D. Binary-Fit

Hint

Apply the stated hole-selection or base-and-bound rule with its explicit convention.

Answer D: Binary-Fit is not one of the standard fit-policy names.

OS28 defines first-fit, next-fit, best-fit and worst-fit, then explicitly says binary-fit is not one of these standard names. First-fit, best-fit and next-fit therefore belong to the allocation-policy category used by the question. Binary-fit is the supported exclusion; it should not be invented as a synonym for another algorithm merely because the word binary is familiar.

Why each choice is right or wrong

A - Incorrect: First-fit selects the first adequate hole and is an explicitly defined memory-allocation policy.

B - Incorrect: Best-fit chooses the smallest adequate hole and is an explicitly defined memory-allocation policy.

C - Incorrect: Next-fit resumes search from the previous position and is an explicitly defined allocation policy.

D - Correct: The note explicitly excludes binary-fit from the standard memory-allocation fit-policy names.

Primary note proof

Study-note section OS28

> First-fit selects the first adequate hole; next-fit continues from the previous search position; best-fit selects the smallest adequate hole; worst-fit selects the largest. No policy universally optimizes both search time and future storage utilization. Binary-fit is not one of these standard names.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 34 | lines 34-45

> External
> fragmentation
> Total memory space is enough to satisfy a request or to reside a
> process in it, but it is not contiguous so it cannot be used.
> 2
> Internal
> fragmentation
> Memory block assigned to process is bigger. Some portion of memory
> is left unused as it cannot be used by another process.
> External fragmentation can be reduced by compaction or shuffle memory contents to place all free memory together in one large block. To make compaction feasible, relocation should be dynamic

Question sources

Operating Systems practice questions · Question 34

Question 74 | Q01317

What is the purpose of a system call interface in an operating system?

A. To manage user accounts

B. To manage hardware resources

C. To manage software applications

D. To provide a standardized interface for application programs to make system calls

Hint

Identify the abstraction layer and separate a protected service request from an application task.

Answer D: Provide the interface through which applications request system calls.

OS04 defines a system call as requesting a protected kernel service and distinguishes an API/library wrapper from the direct boundary. Option D describes the interface that applications use to make those requests. Account, hardware and application management can be services reached through it; they are not the general purpose of the interface itself.

Why each choice is right or wrong

A - Incorrect: Account operations may be exposed as services, but account management does not define the interface itself.

B - Incorrect: Hardware operations may be requested through services; general hardware management is not the interface definition.

C - Incorrect: Application management is a possible service use rather than the purpose of the syscall interface.

D - Correct: The interface supplies the application-to-system-call request path for protected kernel services.

Primary note proof

Study-note section OS04

> A system call requests a protected kernel service, such as creating a process, reading a file or opening a socket. A user-facing API/library may wrap that request; not every library function enters the kernel. Distinguish the application API from the direct user/kernel boundary when describing access to OS services.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 45 | lines 28-33

> Application I/O Interface represents the structuring techniques and interfaces for the operating system to enable I/O devices to be treated in a standard, uniform way. The actual differences lies kernel level modules called device drivers which are custom tailored to corresponding devices but show one of the standard interfaces to applications. The purpose of

the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, such as the I/O system calls. Following are the characteristics of I/O

Operating System module | PDF page 53 | lines 16-19

implement most of the functionalities of the operating system and do not requires kernel module's code access rights.
System Utility - System Utility programs are responsible to do specialized, individual level tasks.

Question sources

Operating Systems practice questions · Question 35

Question 75 | Q01319

What is the purpose of a dispatcher in an operating system?

A. To manage user accounts

B. To manage hardware resources

C. To manage software applications

D. To switch the context of execution from one process to another

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer D: To switch the context of execution from one process to another

PDF p19 describes the dispatcher transferring execution to a queued process; p21 defines the context save/restore mechanism. D is the specific execution handoff rather than a broad management service.

Why each choice is right or wrong

A - Incorrect: Account management does not identify execution handoff.

B - Incorrect: Hardware management is a broad OS responsibility.

C - Incorrect: Application management is not the specific handoff operation.

D - Correct: Switching saved execution contexts matches pp19,21.

Primary note proof

Study-note section OS06

Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work. The dispatcher performs the handoff/context restore after selection.

Study-note section OS05

A context switch saves one execution context and restores another; useful application work is interrupted by this overhead.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 19 | lines 20-25

Non-Running Processes that are not running are kept in queue, waiting for their turn to execute. Each entry in the queue is a pointer to a particular process. Queue is implemented by using linked list. Use of dispatcher is as follows. When a process is interrupted, that process is transferred in the waiting queue. If the process has completed or aborted, the process is discarded. In either case, the dispatcher then selects a process from the queue to execute.

Operating System module | PDF page 21 | lines 41-50

A context switch is the mechanism to store and restore the state or context of a CPU in Process Control block so that a process execution can be resumed from the same point at a later time. Using this technique, a context switcher enables multiple processes to share a single CPU. Context switching is an essential part of a multitasking operating system features.
When the scheduler switches the CPU from executing one process to execute another, the context switcher saves the content of all processor registers for the process being removed from the CPU, in its process descriptor. The context of a process is represented in the process control block of a process. Context switch time is pure overhead. Context switching can significantly affect performance as modern computers have a lot of general

Question sources

Operating Systems practice questions · Question 37

Question 76 | Q01320

Which listed item is not a hardware resource managed by an operating system?

A. CPU

B. Memory

C. Keyboard

D. User account

Hint

Separate broad resource-management goals from individual services or applications.

Answer D: User account

CPU time, memory and input devices such as a keyboard are hardware resources allocated or controlled by the OS. A user account is an identity used in administration and protection, rather than a hardware device. The OS may manage accounts too, but that does not put them in the hardware-resource category. Completing the stem and replacing one option gives a unique exclusion.

Why each choice is right or wrong

A - Incorrect: The CPU is a managed processing resource.

B - Incorrect: Memory is a managed storage resource used during execution.

C - Incorrect: A keyboard is a managed input device.

D - Correct: A user account is an identity rather than a hardware resource.

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs.

Study-note section OS34

> Input/output/storage/NIC devices are OS-managed hardware; user accounts are identities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 5 | lines 6-13

> An operating System (OS) is an intermediary between users and computer hardware. It provides users an environment in which a user can execute programs conveniently and efficiently. In technical terms, it is software which manages hardware. An operating System controls the allocation of resources and services such as memory, processors, devices and information.
> 1.1 Definition
> An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Operating System module | PDF page 43 | lines 7-10

> Computers operate on many kinds of devices. General types include storage devices (disks, tapes), transmission devices (network cards, modems), and human-interface devices (screen, keyboard, mouse). Other devices are more specialized. A device communicates with a computer system by sending signals over a cable or even through the air.

Question sources

Operating Systems practice questions · Question 38

Question 77 | Q01321

39.What is the difference between a process and a program in an operating system?

A. There is no difference between a process and a program

B. A process is a running instance of a program, while a program is a set of instructions

C. A program is a running instance of a process, while a process is a set of instructions

D. A program is a unit of execution, while a process is a unit of scheduling

Hint

Distinguish stored instructions, an executing instance and its saved execution state.

Answer B: A process is a running instance of a program, while a program is a set of instructions

PDF p16 directly contrasts a dynamic executing process with a static program containing processor instructions. B states that distinction.

Why each choice is right or wrong

A - Incorrect: Contradicted by the explicit static/dynamic contrast.

B - Correct: Matches the definitions on p16.

C - Incorrect: Reverses program and process.

D - Incorrect: Program is static instructions, not itself an executing unit (p16).

Primary note proof

Study-note section OS05

> A program is stored instructions; a process is an executing instance with address space and resources. A process does not universally “manage applications” or “provide a user interface.” The PCB records identity, state, PC/registers, scheduling information, memory/accounting details and I/O/open-resource state.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 6-9

> A process is a program in execution. The execution of a process must progress in a sequential fashion. Definition of process is following.
> • A process is defined as an entity which represents the basic unit of work to be implemented in the system.

Operating System module | PDF page 16 | lines 21-24

A program by itself is not a process. It is a static entity made up of program statement while process is a dynamic entity. Program contains the instructions to be executed by processor. A program takes a space at single place in main memory and continues to stay there. A program does not perform any action by itself.

Question sources

Operating Systems practice questions · Question 39

Question 78 | Q01322

Which listed operation can receive bytes from a socket while also serving as a generic file-input operation in the descriptor model?

A. socket()

B. connect()

C. read()

D. send()

Hint

Distinguish a file operation, a resource handle, an offset and file metadata.

Answer C: read()

Sockets can support the ordinary read/write interface as well as specialized socket operations. A read therefore need not be confined to disk files: it can receive bytes from a socket descriptor. socket creates an endpoint, connect establishes a connection, and send transmits data. Only read matches both the generic file-input role and the ability to receive socket data in this model.

Why each choice is right or wrong

A - Incorrect: socket creates a communication endpoint rather than reading bytes from it.

B - Incorrect: connect establishes a connection rather than receiving its data.

C - Correct: read is generic descriptor input and can receive bytes from a socket.

D - Incorrect: send transmits data rather than performing the requested input operation.

Primary note proof

Study-note section OS36

Sockets also support ordinary read/write, alongside socket/connect/send operations, so read is not uniquely “not a network call.” FAT32, NTFS and ext4 are filesystem formats; HTTP is an application communication protocol.

Study-note section OS-S04

In the descriptor-based socket API, socket() creates a communication endpoint and returns its descriptor. connect() associates the socket with the requested peer address; send() transmits data

to another socket. Receiving through read() uses the ordinary descriptor-input interface described in the file-services lesson.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 11 | lines 15-19

• Two processes often require data to be transferred between them.
• The both processes can be on the one computer or on different computer but are connected through computer network.
• Communication may be implemented by two methods either by Shared Memory or by Message Passing.

Operating System module | PDF page 11 | lines 3-8

• Program needs to read a file or write a file.
• The operating system gives the permission to the program for operation on file.
• Permission varies from read-only, read-write, denied and so on.
• Operating System provides an interface to the user to create/delete files.
• Operating System provides an interface to the user to create/delete directories.
• Operating System provides an interface to create the backup of file system.

Question sources

Operating Systems practice questions · Question 40

Question 79 | Q01323

What is the purpose of a system clock in an operating system?

A. To manage user accounts

B. To manage hardware resources

C. To manage software applications

D. To provide a reference for measuring time and scheduling processes and events

Hint

Identify whether data is in transit, retained for reuse or queued for a device.

Answer D: To provide a reference for measuring time and scheduling processes and events

The clock driver maintains time and decrements the quantum counter to invoke the scheduler (PDF p46, section 12.3 Clocks). D is the matching time-and-scheduling purpose.

Why each choice is right or wrong

A - Incorrect: Accounts are not the clock's task.

B - Incorrect: Generic resource management omits the clock's time reference.

C - Incorrect: Software management is not its stated task.

D - Correct: Time of day and quantum expiration are both clock-driver functions on p46.

Primary note proof

Study-note section OS35

> The clock/timer supports timekeeping, timeouts, accounting and preemptive scheduling.

Study-note section OS-S04

> IRQ means interrupt request. A controller can raise an interrupt to request CPU attention. In a clock-tick scheduler, the scheduler initializes a running process's quantum counter; each clock interrupt decrements it, and reaching zero triggers scheduling. Clock support also maintains time of day and usage accounting.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 46 | lines 19-33

> Maintaining the time
> of the day
> The clock driver implements the time of day or the real time
> clock function. It requires incrementing a counter at each
> clock tick.
> 2
> Preventing processes
> from running too long
> As a process is started, the scheduler initializes the quantum
> counter in clock ticks for the process. The clock driver
> decrements the quantum counter by 1, at every clock
> interrupts. When the counter gets to zero, clock driver calls
> the scheduler to set up another process. Thus clock driver
> helps in preventing processes from running longer than time
> slice allowed.

Question sources

Operating Systems practice questions · Question 41

Question 80 | Q01324

42.Which of the following is not a type of device managed by an operating system?

A. Input devices

B. Output devices

C. Storage devices

D. User accounts

Hint

Separate device hardware, device-specific software and general event or transfer mechanisms.

Answer D: User accounts

PDF p43 lists storage, transmission and human-interface hardware. User accounts concern authentication (p50), rather than physical I/O device categories.

Why each choice is right or wrong

A - Incorrect: Human input devices are listed (p43).

B - Incorrect: Screens and related human-interface devices are listed (p43).

C - Incorrect: Disks and tapes are storage devices (p43).

D - Correct: A user identity/account is authenticated, not a physical device (p50).

Primary note proof

Study-note section OS34

> Input/output/storage/NIC devices are OS-managed hardware; user accounts are identities.

Study-note section OS39

> Authentication verifies identity; authorization checks permitted operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 43 | lines 7-10

> Computers operate on many kinds of devices. General types include storage devices (disks, tapes), transmission devices (network cards, modems), and human-interface devices (screen, keyboard, mouse). Other devices are more specialized. A device communicates with a computer system by sending signals over a cable or even through the air.

Operating System module | PDF page 50 | lines 18-28

> Authentication refers to identifying each user of the system and associating the executing programs with those users. It is the responsibility of the Operating System to create a protection system which ensures that a user who is running a particular program is authentic. Operating Systems generally identifies/authenticates users using following three ways:
> Username / Password - User need to enter a registered username and password with Operating system to login into the system.
> User card/key - User need to punch card in card slot, or enter key generated by key generator in option provided by operating system to login into the system.
> User attribute - fingerprint/ eye retina pattern/ signature - User need to pass his/her attribute via designated input device used by operating system to login into the system.

Question sources

Operating Systems practice questions · Question 42

Question 81 | Q01325

What is the purpose of a file descriptor in an operating system?

A. To manage user accounts

B. To manage hardware resources

C. To manage software applications

D. To provide a reference to an open file or communication endpoint

Hint

Distinguish a file operation, a resource handle, an offset and file metadata.

Answer D: A process-local reference to an open file or communication endpoint.

OS36 defines a file descriptor as a process-local integer handle and separately places offset and access mode in associated open-file state. It also describes sockets supporting ordinary descriptor I/O. The descriptor therefore refers to an open resource or endpoint; it is not an account manager, hardware manager or application manager. D is the specific role matching that definition.

Why each choice is right or wrong

A - Incorrect: A descriptor identifies an open resource within a process, rather than managing user accounts.

B - Incorrect: A descriptor is a handle used to access a resource, not a general hardware-management component.

C - Incorrect: Descriptors support resource I/O and do not manage application lifecycles or software installations.

D - Correct: An open-file or socket reference is the resource-handle purpose of a file descriptor.

Primary note proof

Study-note section OS36

> A file has content and metadata such as name, size, owner, permissions and timestamps. open obtains an open-file reference, read/write transfer bytes, seek changes an offset, close releases a reference, truncate changes size and unlink removes a name. A file descriptor is a process-local integer handle; the associated open-file state contains the current offset and access mode.

Study-note section OS36

> Sockets also support ordinary read/write, alongside socket/connect/send operations, so read is not uniquely “not a network call.” FAT32, NTFS and ext4 are filesystem formats; HTTP is an application communication protocol.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 48 | lines 5-8

13.1 File
A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the file's creator and user.

Operating System module | PDF page 11 | lines 3-8

- Program needs to read a file or write a file.
- The operating system gives the permission to the program for operation on file.
- Permission varies from read-only, read-write, denied and so on.
- Operating System provides an interface to the user to create/delete files.
- Operating System provides an interface to the user to create/delete directories.
- Operating System provides an interface to create the backup of file system.

Question sources

Operating Systems practice questions · Question 43

Question 82 | Q01454

What is the main function of the command interpreter?

A. to provide the interface between the API and application program

B. to handle the files in the operating system

C. to get and execute the next user-specified command

D. none of the mentioned

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer C: to get and execute the next user-specified command

The shell is a special interpreter that executes OS commands and application programs (PDF p54), taking user commands to kernel functions. C states that command-interpreter loop role.

Why each choice is right or wrong

A - Incorrect: The module places the shell between user commands and kernel, not API and application.

B - Incorrect: File handling is one possible command activity, not the interpreter's general function.

C - Correct: Getting and executing user commands matches p54.

D - Incorrect: C already matches the supplied definition.

Primary note proof

Study-note section OS03

The kernel implements core resource management, protection and system-call handling. A shell reads/interprets user commands; a GUI is another interface. Browsers, office tools and editors are applications.

Study-note section OS04

A user-facing API/library may wrap that request; not every library function enters the kernel.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 54 | lines 18-20

commands of the operating system. It can be used to do various types of operations, call application programs etc.
Security - Linux provides user security using authentication features like password

Operating System module | PDF page 54 | lines 24-30

Hardware layer - Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc.).
Kernel - Core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.
Shell - An interface to kernel, hiding complexity of kernel's functions from users. Takes commands from user and executes kernel's functions.
Utilities - Utility programs giving user most of the functionalities of an operating systems.

Question sources

Operating Systems practice questions · Question 2

Question 83 | Q01458

Which failures can OS software report or handle while it continues executing on powered hardware? C means complete power loss with no backup supply or advance notification. Select all that apply.

A. lack of paper in printer

B. connection failure in the network

C. power failure

D. all of the mentioned

Hint

Separate device hardware, device-specific software and general event or transfer mechanisms.

Answer A, B: lack of paper in printer; connection failure in the network

OS I/O services can report or respond to a printer running out of paper and to a network connection failure. Complete unannounced loss of power without backup prevents the processor from continuing to run the OS, so software alone cannot handle it during continued execution. The aggregate option includes that unsupported case. Under the explicit power assumptions, the printer and network choices are selected.

Why each choice is right or wrong

A - Correct: The OS can report or handle an out-of-paper printer condition while running.

B - Correct: The OS can report or handle a network connection failure while running.

C - Incorrect: Complete unannounced power loss without backup prevents continued software execution.

D - Incorrect: All includes the excluded total-power-loss case and is therefore incorrect.

Primary note proof

Study-note section OS34

> Printer-paper and network errors can be handled/reported, but continued operation during total power loss requires hardware support, not OS software alone.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 47 | lines 17-18

> Error Handling - An operating system that uses protected memory can guard against many kinds of hardware and application errors.

Operating System module | PDF page 45 | lines 10-22

> The CPU hardware uses an interrupt request line wire which helps CPU to sense after executing every instruction. When the CPU checks that a controller has put a signal on the interrupt request line, the CPU saves a state, such as the current value of the instruction pointer, and jumps to the interrupt handler routine at a fixed address. The interrupt handler part determines the cause of the interrupt performs the necessary processing and executes an interrupt instruction to return the CPU to its execution state.
> The basic mechanism of interrupt enables the CPU to respond to an asynchronous event, such as when a device controller becomes ready for service. Most CPUs have two interrupt request lines.
> • non-maskable interrupt - Such kind of interrupts are reserved for events like unrecoverable memory errors.
> • maskable interrupt - Such interrupts can be switched off by the CPU before the execution of critical instructions that must not be interrupted.

Question sources

Operating Systems practice questions · Question 6

Question 84 | Q01459

Who among the following is credited as a co-creator of the UNIX operating system at Bell Labs, together with Ken Thompson?

A. Charles Babbage

B. Alan Turing

C. Tim Berners-Lee

D. Dennis Ritchie

Hint

Compare the stated historical clues and distinguish origin, characteristic era and technical capability.

Answer D: Dennis Ritchie

Dennis Ritchie is credited alongside Ken Thompson in Bell Labs' account of UNIX's creation. The other listed people are not that co-creator.

Why each choice is right or wrong

A - Incorrect: Babbage is not the UNIX co-creator alongside Thompson identified by Bell Labs.

B - Incorrect: Turing is not the UNIX co-creator alongside Thompson identified by Bell Labs.

C - Incorrect: Berners-Lee is not the UNIX co-creator alongside Thompson identified by Bell Labs.

D - Correct: Dennis Ritchie is the Bell Labs co-creator associated with Ken Thompson.

Primary note proof

Study-note section OS-S01

> UNIX development began at Bell Labs in 1969, led by Ken Thompson and Dennis Ritchie with other collaborators. Ritchie is therefore a co-creator, not the sole inventor of the first operating system.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 5-7

> Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

Question sources

Operating Systems practice questions · Question 7

Question 85 | Q01460

Which operating system was developed by Microsoft as a successor to MS-DOS?

A. macOS

B. Windows

C. Linux

D. Solaris

Hint

Use the stated historical or platform scope and distinguish origin from technical capability.

Answer B: Windows.

OS40 explicitly identifies Windows as the Microsoft family commonly described as succeeding the MS-DOS desktop era. That directly matches the stem's company and predecessor clues. Linux is separately attributed to Linus Torvalds, Solaris to Sun, and macOS is a different proprietary OS product. No additional market-history claim is needed to resolve the listed alternatives.

Why each choice is right or wrong

A - Incorrect: macOS is not the Microsoft family identified as succeeding the MS-DOS desktop era.

B - Correct: The note explicitly identifies Windows as Microsoft's successor family to the MS-DOS desktop era.

C - Incorrect: Linux is separately attributed to Linus Torvalds, not the Microsoft MS-DOS desktop succession.

D - Incorrect: Solaris originated at Sun rather than being Microsoft's MS-DOS successor family.

Primary note proof

Study-note section OS40

> Windows is the Microsoft family commonly described as succeeding the MS-DOS desktop era; Solaris originated at Sun.

Study-note section OS40

> Linux is a Unix-compatible kernel written from scratch by Linus Torvalds; a distribution combines it with userland, libraries and tools.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 8 | lines 21-26

> Network Operating System runs on a server and provides server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or

to other networks. Examples of network operating systems are Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.

Question sources

Operating Systems practice questions · Question 8

Question 86 | Q01461

Which listed operating-system family originated at Bell Labs in 1969 and later provided multiuser time-sharing with asynchronous processes? This asks for that system's identity, not the first-ever invention of multitasking.

A. MS-DOS

B. macOS

C. Windows

D. UNIX

Hint

Compare the stated historical clues and distinguish origin, characteristic era and technical capability.

Answer D: UNIX

UNIX began at Bell Labs in 1969. Ritchie and Thompson's primary paper describes the later system as multiuser and interactive, with asynchronous processes. These historical identifying facts come from the supplementary sources. OS40 supplies the related caution that UNIX was not the first system ever to multitask. The revised question preserves the history and multitasking focus while dropping that false priority claim.

Why each choice is right or wrong

A - Incorrect: MS-DOS is not the Bell Labs system originating in 1969.

B - Incorrect: macOS is not the original Bell Labs system identified by that history.

C - Incorrect: Windows is not the Bell Labs system described by Ritchie and Thompson.

D - Correct: UNIX matches the Bell Labs origin and the later multiuser time-sharing description.

Primary note proof

Study-note section OS-S01

UNIX development began at Bell Labs in 1969, led by Ken Thompson and Dennis Ritchie with other collaborators. Ritchie is therefore a co-creator, not the sole inventor of the first operating system.

Study-note section OS-S01

The later UNIX time-sharing system described by Ritchie and Thompson is multiuser and interactive and supports asynchronous processes. Its later versions incorporated multiprogramming. This history does not make UNIX the first system ever to multitask.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 13 | lines 21-25

4.3 Multiprogramming
When two or more programs are residing in memory at the same time, then sharing the processor is referred to the multiprogramming. Multiprogramming assumes a single shared processor. Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has one to execute.

Operating System module | PDF page 7 | lines 20-33

Time sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multi-programmed Batch Systems and Time-Sharing Systems is that in case of multi-programmed batch systems, objective is to maximize processor use, whereas in Time-Sharing Systems objective is to minimize response time.
Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, processor execute each user program in a short burst or quantum of computation. That is if n users are present, each user can get time quantum. When the user submits the command, the response time is in few seconds at most.
Operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems

Question sources

Operating Systems practice questions · Question 9

Question 87 | Q01462

Which operating system was developed by Linus Torvalds and is based on the UNIX operating system?

A. Linux

B. Windows

C. macOS

D. Solaris

Hint

Use the stated historical or platform scope and distinguish origin from technical capability.

Answer A: Linux, Unix-compatible and written from scratch.

OS40 attributes Linux to Linus Torvalds and calls it a Unix-compatible kernel written from scratch. The developer clue therefore uniquely identifies Linux. The phrase based on UNIX must be read as compatible/inspired in interface and design, rather than asserting copied UNIX source. The other named families do not match the explicit creator attribution given here.

Why each choice is right or wrong

A - Correct: Linux matches Linus Torvalds and Unix compatibility; the note explicitly says it was written from scratch.

B - Incorrect: Windows is the Microsoft family, not the kernel explicitly attributed here to Linus Torvalds.

C - Incorrect: macOS does not match the note's explicit Torvalds attribution for the Unix-compatible kernel.

D - Incorrect: Solaris originated at Sun rather than being the kernel developed by Linus Torvalds.

Primary note proof

Study-note section OS40

> Linux is a Unix-compatible kernel written from scratch by Linus Torvalds; a distribution combines it with userland, libraries and tools.

Study-note section OS40

> Windows is the Microsoft family commonly described as succeeding the MS-DOS desktop era; Solaris originated at Sun.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 5-7

> Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

Question sources

Operating Systems practice questions · Question 10

Question 88 | Q01463

Which operating system was developed by Sun Microsystems and is known for its scalability and reliability?

A. Linux

B. Windows

C. macOS

D. Solaris

Hint

Use the stated historical or platform scope and distinguish origin from technical capability.

Answer D: Solaris.

OS40 directly states that Solaris originated at Sun. That developer clue resolves the choices even though the note does not quantify Solaris's reliability or scalability relative to other systems. The answer relies on the explicit origin, not on assuming those broad positive attributes uniquely distinguish one OS. Linux and Windows have separately identified developer lineages in the same section.

Why each choice is right or wrong

A - Incorrect: Linux is attributed to Linus Torvalds rather than being the Sun-originated OS requested here.

B - Incorrect: Windows is identified as the Microsoft family, not the Sun-originated product.

C - Incorrect: macOS does not match the explicit Sun-origin attribution provided by the complete note.

D - Correct: Solaris is explicitly stated to have originated at Sun, matching the decisive developer clue.

Primary note proof

Study-note section OS40

> Windows is the Microsoft family commonly described as succeeding the MS-DOS desktop era; Solaris originated at Sun.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 29 | lines 4-16

> In this case, thread management done by the Kernel. There is no thread management code in the application area. Kernel threads are supported directly by the operating system. Any application can be programmed to be multithreaded. All of the threads within an application are supported within a single process.
> The Kernel maintains context information for the process as a whole and for individuals' threads within the process. Scheduling by the Kernel is done on a thread basis. The Kernel

performs thread creation, scheduling and management in Kernel space. Kernel threads are generally slower to create and manage than the user threads.
Advantages
• Kernel can simultaneously schedule multiple threads from the same process on multiple processes.
• If one thread in a process is blocked, the Kernel can schedule another thread of the same process.

Question sources

Operating Systems practice questions · Question 11

Question 89 | Q01464

Which listed mobile operating-system platform uses Linux as its base kernel layer and has an open-source base called AOSP?

A. iOS

B. Android

C. Windows Phone

D. BlackBerry OS

Hint

Use the stated historical or platform scope and distinguish origin from technical capability.

Answer B: Android

Android uses Linux at its base kernel layer, with framework and application layers above it. Its open-source base is AOSP, although shipped devices may contain proprietary services or applications. Those architectural facts identify Android without needing an unspecified population or a dated market-share claim. The other listed mobile systems are not the platform identified by this particular combination of Linux and AOSP.

Why each choice is right or wrong

A - Incorrect: iOS is not the platform identified by the Linux/AOSP combination.

B - Correct: Android has the Linux base layer and the AOSP open-source base.

C - Incorrect: Windows Phone is not the Linux/AOSP platform described.

D - Incorrect: BlackBerry OS is not the Linux/AOSP platform described.

Primary note proof

Study-note section OS40

Android’s base kernel layer is Linux; its framework and applications are higher layers. AOSP is open source, while a shipped device can include proprietary applications/services.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 8 | lines 21-26

> Network Operating System runs on a server and provides server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks. Examples of network operating systems are Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.

Question sources

Operating Systems practice questions · Question 12

Question 90 | Q01471

Using the first four generations described in NYU's OS-history notes (vacuum tubes; transistors; integrated circuits; personal computers), which generation is associated with resident uniprogrammed batch operating systems, following the manual/no-resident-OS era?

A. First Generation

B. Second Generation

C. Third Generation

D. Fourth Generation

Hint

Compare the stated historical clues and distinguish origin, characteristic era and technical capability.

Answer B: Second Generation

The first four generations in NYU's historical taxonomy associates second-generation transistor machines with uniprogrammed batch systems whose OS remains resident between jobs. The first era has no separate resident OS; the third emphasizes ICs and multiprogramming, and the fourth personal computers. OS02 explains batch operation, while the generation mapping is supplied by the supplementary university notes. This is a scoped teaching convention rather than a universal first-OS claim.

Why each choice is right or wrong

A - Incorrect: The first era is the manual/no-separate-resident-OS era in this taxonomy.

B - Correct: The second era pairs transistor machines with resident uniprogrammed batch systems.

C - Incorrect: The third era emphasizes ICs and multiprogramming rather than the initial resident batch stage.

D - Incorrect: The fourth era emphasizes personal computers and follows the batch-monitor era.

Primary note proof

Study-note section OS-S01

> In the first four generations of the cited NYU teaching taxonomy, the first generation uses vacuum tubes and manual single-user operation without a separate resident OS. The second generation pairs transistors with uniprogrammed batch systems; the resident OS starts successive jobs without being reloaded for every job.

Study-note section OS-S01

> The third generation pairs integrated circuits with multiprogramming. The fourth emphasizes personal computers. These are characteristic-era associations, not claims that each feature existed exclusively in one era.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 7 | lines 20-33

> Time sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multi-programmed Batch Systems and Time-Sharing Systems is that in case of multi-programmed batch systems, objective is to maximize processor use, whereas in Time-Sharing Systems objective is to minimize response time.
> Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, processor execute each user program in a short burst or quantum of computation. That is if n users are present, each user can get time quantum. When the user submits the command, the response time is in few seconds at most.
> Operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems

Operating System module | PDF page 7 | lines 5-7

> operating systems are there from the very first computer generation. Operating systems keep evolving over the period of time. Following are few of the important types of operating system which are most commonly used.

Question sources

Operating Systems practice questions · Question 19

Question 91 | Q01475

What is the primary function of a kernel in an operating system?

A. Provide a user-friendly interface

B. Manage system resources and provide core services

C. Run user applications

D. Facilitate communication with external devices

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer B: Manage system resources and provide core services

The kernel is the core, responsible for major activities and direct hardware interaction (PDF p53), providing services with privileged access. B summarizes this rather than a particular interface or device subset.

Why each choice is right or wrong

A - Incorrect: The shell supplies the user-command interface (p54).

B - Correct: Core services and resource access match p53.

C - Incorrect: The kernel runs processes, but application execution alone omits its wider central role.

D - Incorrect: External-device communication is one hardware activity, not the entire primary role.

Primary note proof

Study-note section OS03

> The kernel implements core resource management, protection and system-call handling. A shell reads/interprets user commands; a GUI is another interface. Browsers, office tools and editors are applications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 11-15

> operating system. It is consisting of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
> System Library - System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries

Operating System module | PDF page 53 | lines 21-29

> Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single

address space and do not require any context switch and hence is very efficient and fast.
Kernel runs each process and provides system services to processes, provides protected access to hardware to processes.
Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

Operating System module | PDF page 54 | lines 18-20

commands of the operating system. It can be used to do various types of operations, call application programs etc.
Security - Linux provides user security using authentication features like password

Question sources

Operating Systems practice questions · Question 23

Question 92 | Q01476

Which component of an operating system is responsible for memory management and process scheduling?

A. Kernel

B. User program

C. Compiler

D. File system

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer A: Kernel

Kernel is the major-activity core (PDF p53); memory management and ready-process selection are OS functions (pp31,20). Under the module's architecture, A is the component responsible for these core services.

Why each choice is right or wrong

A - Correct: Kernel provides the core resource services (p53).

B - Incorrect: User programs request services rather than being the core manager (p53).

C - Incorrect: Compilation is not the ready-process or memory manager described.

D - Incorrect: File organization does not substitute for memory/CPU management.

Primary note proof

Study-note section OS03

The kernel implements core resource management, protection and system-call handling.

Study-note section OS01

An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs.

Study-note section OS06

Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Study-note section OS-S05

A compiler translates source code toward an executable representation. For example, the conventional GCC toolchain distinguishes preprocessing, compilation proper, assembly and linking. A compiler uses operating-system services as a development tool; it is not the runtime kernel component that selects ready processes, manages memory or handles protected service requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 11-15

operating system. It is consisting of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
System Library - System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries

Operating System module | PDF page 31 | lines 24-31

Memory management is the functionality of an operating system which handles or manages primary memory. Memory management keeps track of each and every memory location either it is allocated to some process or it is free. It checks how much memory is to be allocated to processes. It decides which process will get memory at what time. It tracks whenever some memory gets freed or unallocated and correspondingly it updates the status.
Memory management provides protection by using two registers, a base register and a limit register. The base register holds the smallest legal physical memory address and the limit register specifies the size of the range. For example, if the base register holds 300000 and the

Operating System module | PDF page 20 | lines 15-21

It is also called CPU scheduler. Main objective is increasing system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects process among the processes that are ready to execute and allocates CPU to one of them.
Short term scheduler also known as dispatcher, execute most frequently and makes the fine-grained decision of which process to execute next. Short term scheduler is faster than long term scheduler.

Question sources

Operating Systems practice questions · Question 24

Question 93 | Q01477

In the module’s conventional architecture, where device drivers are kernel components and ordinary applications run with restricted privilege, which item normally executes in kernel mode?

A. User programs

B. Device drivers

C. Text editors

D. Web browsers

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer B: Device drivers

Kernel components perform privileged operating-system work. In the stated architecture, device drivers are such components, so they execute in kernel mode. User programs, text editors and web browsers are ordinary applications and execute in restricted user mode, requesting protected services when needed. User-space driver designs exist, which is why the architecture is stated rather than treating every driver implementation as privileged.

Why each choice is right or wrong

A - Incorrect: Ordinary user programs execute in restricted user mode.

B - Correct: The specified drivers are kernel components and therefore execute with kernel privilege.

C - Incorrect: A text editor is an ordinary application, so it runs in user mode under this architecture.

D - Incorrect: A web browser is an ordinary application and does not acquire kernel privilege merely by using OS services.

Primary note proof

Study-note section OS03

> Browsers, office tools and editors are applications. Device drivers are commonly kernel components, although user-space drivers also exist.

Study-note section OS-S02

> Being scheduled or managed by the kernel does not give application thread code kernel privilege. Ordinary applications and supporting utilities execute in restricted user mode and request protected services through libraries and system calls. Kernel component code executes in privileged kernel mode and performs permitted low-level hardware operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 45 | lines 28-33

> Application I/O Interface represents the structuring techniques and interfaces for the operating system to enable I/O devices to be treated in a standard, uniform way. The actual differences lies kernel level modules called device drivers which are custom tailored to corresponding devices but show one of the standard interfaces to applications. The purpose of the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, such as the I/O system calls. Following are the characteristics of I/O

Operating System module | PDF page 53 | lines 21-29

> Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast. Kernel runs each process and provides system services to processes, provides protected access to hardware to processes.
> Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

Question sources

Operating Systems practice questions · Question 25

Question 94 | Q01478

In the operating system’s user/kernel privilege model, which standard term names the restricted level used by ordinary application code? Classify execution privilege, rather than processor addressing modes.

A. Kernel mode

B. Supervisor mode

C. User mode

D. Protected mode

Hint

Distinguish where threads are managed from the privilege of the code they execute.

Answer C: User mode

Ordinary application code runs at the restricted user level and obtains protected services through libraries and system calls. Kernel mode is the privileged level; supervisor mode is commonly another name for that privileged level. Protected mode is not the name of the restricted application privilege level in the stated user/kernel classification. The question therefore selects User mode without making a claim about every processor’s architectural modes.

Why each choice is right or wrong

A - Incorrect: Kernel mode permits privileged OS operations and is not the restricted application level.

B - Incorrect: Supervisor mode denotes the privileged context in this model, rather than ordinary application privilege.

C - Correct: User mode is the restricted application level in the user/kernel privilege model.

D - Incorrect: The requested restricted privilege level is user mode; an architectural mode label does not name that level in this classification.

Primary note proof

Study-note section OS-S02

> Being scheduled or managed by the kernel does not give application thread code kernel privilege. Ordinary applications and supporting utilities execute in restricted user mode and request protected services through libraries and system calls. Kernel component code executes in privileged kernel mode and performs permitted low-level hardware operations.

Study-note section OS03

> Kernel mode has privileged operations; supervisor mode often means the same privileged mode, so offering both as exclusive answers is ambiguous.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 21-29

> Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast.
> Kernel runs each process and provides system services to processes, provides protected access to hardware to processes.
> Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

Question sources

Operating Systems practice questions · Question 26

Question 95 | Q01479

In the teaching module’s architecture, which broad core component provides protected file-system operations for applications?

A. Kernel

B. User program

C. Device driver

D. Compiler

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer A: Kernel

The kernel provides core protected services and handles system calls, including the operating system’s file operations in this architecture. Applications request those services. A device driver translates requests into device-specific operations beneath the file-service layer, while a compiler translates program source. These supporting roles do not make either the broad core provider of protected file-system operations.

Why each choice is right or wrong

A - Correct: The kernel provides the protected OS service and system-call layer in the stated architecture.

B - Incorrect: Applications request file services rather than being the protected core service provider.

C - Incorrect: Drivers perform device-specific operations beneath the broader file-service layer.

D - Incorrect: A compiler translates program source; it is not the runtime protected file-service provider.

Primary note proof

Study-note section OS03

> The kernel implements core resource management, protection and system-call handling.

Study-note section OS36

> A file has content and metadata such as name, size, owner, permissions and timestamps. open obtains an open-file reference, read/write transfer bytes, seek changes an offset, close releases a reference, truncate changes size and unlink removes a name.

Study-note section OS34

> A driver converts OS requests into device-specific operations.

Study-note section OS-S05

> A compiler translates source code toward an executable representation. For example, the conventional GCC toolchain distinguishes preprocessing, compilation proper, assembly and

linking. A compiler uses operating-system services as a development tool; it is not the runtime kernel component that selects ready processes, manages memory or handles protected service requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 11-15

operating system. It is consisting of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
System Library - System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries

Operating System module | PDF page 6 | lines 19-23

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Operating System does the following activities for file management.
Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.

Operating System module | PDF page 45 | lines 28-33

Application I/O Interface represents the structuring techniques and interfaces for the operating system to enable I/O devices to be treated in a standard, uniform way. The actual differences lies kernel level modules called device drivers which are custom tailored to corresponding devices but show one of the standard interfaces to applications. The purpose of the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, such as the I/O system calls. Following are the characteristics of I/O

Question sources

Operating Systems practice questions · Question 27

Question 96 | Q01480

During a conventional disk-based boot, the bootloader reads the kernel image from disk. Into which main execution-memory region is it loaded before control is transferred to it?

A. RAM

B. Hard disk

C. CPU cache

D. ROM

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer A: RAM

In the stated boot sequence, firmware and bootloader stages run before control passes to the kernel loaded into RAM. The disk supplies the stored image, while RAM is its main execution-memory destination. CPU cache can retain copies of memory contents for faster reuse, but that is distinct from the image's main-memory loading location. ROM is not the RAM destination specified by this conventional boot model.

Why each choice is right or wrong

A - Correct: RAM is the main execution-memory destination for the loaded kernel image in the stated boot sequence.

B - Incorrect: The hard disk is the source of the stored image here, not its main-memory loading destination.

C - Incorrect: Cache retains copies for reuse; it is not the bootloader's main-memory image region.

D - Incorrect: ROM does not identify the writable RAM destination of the kernel image in this stated disk-boot model.

Primary note proof

Study-note section OS03

> Firmware/bootloader stages execute before transferring control to a kernel loaded into RAM.

Study-note section OS35

> Caching retains copies to accelerate reuse.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 5 | lines 26-36

> Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address. Main memory provides a fast storage that can be access directly by the CPU. So, for a program to be executed, must in the main memory.
> Activities Operating System for memory management.
> Keeps tracks of primary memory (i.e. what part of it are in use by whom, what part are not in use.)
> In multiprogramming, OS decides which process will get memory when and how much.
> Allocates the memory when the process requests it to do so.
> De-allocates the memory when the process no longer needs it or has been terminated.

Operating System module | PDF page 31 | lines 24-31

> Memory management is the functionality of an operating system which handles or manages primary memory. Memory management keeps track of each and every memory location either it is allocated to some process or it is free. It checks how much memory is to be allocated to processes. It decides which process will get memory at what time. It tracks whenever some

memory gets freed or unallocated and correspondingly it updates the status.
Memory management provides protection by using two registers, a base register and a limit register. The base register holds the smallest legal physical memory address and the limit register specifies the size of the range. For example, if the base register holds 300000 and the

Operating System module | PDF page 33 | lines 25-28

9.5 Memory Allocation
Main memory usually has two partitions
• Low Memory -- Operating system resides in this memory.
• High Memory -- User processes then held in high memory.

Question sources

Operating Systems practice questions · Question 28

Question 97 | Q01481

In a conventional monolithic OS, which listed utilities can have user-space program components that invoke privileged OS services, even if the complete product also includes privileged helpers? Treat “device driver” and “interrupt handler” here as kernel-resident components. Select all that apply.

A. Device driver

B. Antivirus software

C. Disk partition manager

D. Interrupt handler

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer B, C: Antivirus software; Disk partition manager

Antivirus and disk-management products can combine ordinary user processes with privileged helper components. Their user-space components count as user programs even though some work requires kernel services. The stem explicitly confines the driver and interrupt-handler options to kernel-resident components, avoiding architectures with user-space drivers. Thus antivirus software and the partition-manager utility are both valid examples under the stated component-level interpretation.

Why each choice is right or wrong

A - Incorrect: The driver is explicitly kernel-resident in this question.

B - Correct: Antivirus software can have a user-process component alongside privileged helpers.

C - Correct: A disk partition manager can have a user-process component invoking privileged services.

D - Incorrect: The interrupt handler is explicitly kernel-resident in this question.

Primary note proof

Study-note section OS03

> Utilities such as antivirus and disk tools can combine user processes with privileged components.

Study-note section OS-S02

> Being scheduled or managed by the kernel does not give application thread code kernel privilege. Ordinary applications and supporting utilities execute in restricted user mode and request protected services through libraries and system calls. Kernel component code executes in privileged kernel mode and performs permitted low-level hardware operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 21-29

> Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast.
> Kernel runs each process and provides system services to processes, provides protected access to hardware to processes.
> Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

Operating System module | PDF page 47 | lines 19-26

> 12.5 Device driver
> Device driver is a program or routine developed for an I/O device. A device driver implements I/O operations or behaviors on a specific class of devices. For example a system supports one or a number of multiple brands of terminals, all slightly different terminals may have a single terminal driver. In the layered structure of I/O system, device driver lies between interrupt handler and device independent I/O software. The job of a device driver is following.
> To accept request from the device independent software above it.
> To see to it that the request is executed.

Operating System module | PDF page 45 | lines 10-22

> The CPU hardware uses an interrupt request line wire which helps CPU to sense after executing every instruction. When the CPU checks that a controller has put a signal on the interrupt request line, the CPU saves a state, such as the current value of the instruction pointer, and jumps to the interrupt handler routine at a fixed address. The interrupt handler part determines the cause of the interrupt performs the necessary processing and executes an interrupt instruction to return the CPU to its execution state.
> The basic mechanism of interrupt enables the CPU to respond to an asynchronous event, such as when a device controller becomes ready for service. Most CPUs have two interrupt request lines.

• non-maskable interrupt - Such kind of interrupts are reserved for events like unrecoverable memory errors.
• maskable interrupt - Such interrupts can be switched off by the CPU before the execution of critical instructions that must not be interrupted.

Question sources

Operating Systems practice questions · Question 29

Question 98 | Q01482

In the module's operating-system architecture, which broad core component performs privileged interaction with underlying hardware?

A. Kernel

B. User program

C. Compiler

D. File system

Hint

Distinguish where threads are managed from the privilege of the code they execute.

Answer A: Kernel

The kernel performs privileged low-level hardware operations and supplies protected services. Ordinary applications use those services from user mode. A compiler translates source code, and the file system provides a particular family of storage services rather than naming the broad hardware-facing OS core. The question asks for that broad core component, not a particular device-specific driver.

Why each choice is right or wrong

A - Correct: The kernel is the privileged OS core that performs permitted low-level hardware operations.

B - Incorrect: Ordinary user programs request protected services rather than directly obtaining kernel privilege.

C - Incorrect: A compiler performs source translation, a development-tool role rather than privileged OS-core hardware management.

D - Incorrect: A file system supplies specific storage services, not the broad hardware-facing OS core requested.

Primary note proof

Study-note section OS-S02

Being scheduled or managed by the kernel does not give application thread code kernel privilege. Ordinary applications and supporting utilities execute in restricted user mode and

request protected services through libraries and system calls. Kernel component code executes in privileged kernel mode and performs permitted low-level hardware operations.

Study-note section OS-S05

A compiler translates source code toward an executable representation. For example, the conventional GCC toolchain distinguishes preprocessing, compilation proper, assembly and linking. A compiler uses operating-system services as a development tool; it is not the runtime kernel component that selects ready processes, manages memory or handles protected service requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 11-15

operating system. It is consisting of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
System Library - System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries

Operating System module | PDF page 53 | lines 21-29

Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast.
Kernel runs each process and provides system services to processes, provides protected access to hardware to processes.
Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

Question sources

Operating Systems practice questions · Question 30

Question 99 | Q01483

In a two-level privilege model where “kernel mode” and “supervisor mode” are synonymous names for the privileged execution level, which listed names denote that level? “Protected mode” denotes a processor operating mode that can include restricted user execution. Select all that apply.

A. Kernel mode

B. User mode

C. Supervisor mode

D. Protected mode

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer A, C: Kernel mode; Supervisor mode

The note states that kernel mode permits privileged operations and that supervisor mode commonly denotes the same privileged mode. The stated model makes those names explicitly synonymous. User mode is restricted. Protected mode is defined here as a broader processor mode that can still run user code, so its name alone does not identify the privileged execution level. Both kernel and supervisor are selected.

Why each choice is right or wrong

A - Correct: Kernel mode denotes the privileged execution level.

B - Incorrect: User mode is the restricted level.

C - Correct: Supervisor mode is explicitly synonymous with kernel mode in this model.

D - Incorrect: Protected mode is not a privilege level under the supplied definition.

Primary note proof

Study-note section OS03

> Kernel mode has privileged operations; supervisor mode often means the same privileged mode, so offering both as exclusive answers is ambiguous.

Study-note section OS-S02

> Being scheduled or managed by the kernel does not give application thread code kernel privilege. Ordinary applications and supporting utilities execute in restricted user mode and request protected services through libraries and system calls. Kernel component code executes in privileged kernel mode and performs permitted low-level hardware operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 21-29

> Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast.
> Kernel runs each process and provides system services to processes, provides protected access to hardware to processes.
> Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

Question sources

Operating Systems practice questions · Question 31

Question 100 | Q01484

Which of the following is responsible for handling system calls made by user programs?

A. Kernel

B. User program

C. Compiler

D. File system

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer A: Kernel

PDF p53 says the kernel provides system services and protected hardware access to processes; libraries let user code reach kernel functions. Thus the handling component is the kernel, not the application caller or compiler.

Why each choice is right or wrong

A - Correct: Kernel provides the requested process services (p53).

B - Incorrect: User program is the requester using libraries (p53).

C - Incorrect: Compiler is not the runtime service handler.

D - Incorrect: File system handles one service category, not all kernel requests.

Primary note proof

Study-note section OS03

> The kernel implements core resource management, protection and system-call handling.

Study-note section OS04

> A system call requests a protected kernel service, such as creating a process, reading a file or opening a socket. A user-facing API/library may wrap that request; not every library function enters the kernel.

Study-note section OS-S05

> A compiler translates source code toward an executable representation. For example, the conventional GCC toolchain distinguishes preprocessing, compilation proper, assembly and linking. A compiler uses operating-system services as a development tool; it is not the runtime kernel component that selects ready processes, manages memory or handles protected service requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 21-29

> Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast.
> Kernel runs each process and provides system services to processes, provides protected access to hardware to processes.
> Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

Operating System module | PDF page 53 | lines 16-19

> implement most of the functionalities of the operating system and do not requires kernel module's code access rights.
> System Utility - System Utility programs are responsible to do specialized, individual level tasks.

Question sources

Operating Systems practice questions · Question 32

Question 101 | Q01506

A scheduler selects the next process to be executed from one of the following.

A. Ready queue

B. Job queue

C. Device queue

D. None of the choices is correct

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer A: Ready queue

PDF pp19-20 place processes waiting for CPU in the ready queue and select among ready processes. The question is about the next CPU execution, hence A rather than job admission.

Why each choice is right or wrong

A - Correct: Ready queue contains eligible CPU waiters.

B - Incorrect: Job queue includes all processes/admission work, not just eligible CPU contenders.

C - Incorrect: Device queue contains I/O waiters (p18).

D - Incorrect: Ready queue is explicitly supplied.

Primary note proof

Study-note section OS06

The ready queue contains eligible work; device queues hold blocked requests. The broad textbook job queue contains processes in the system, while an admission queue more narrowly holds jobs awaiting memory/admission. Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 19 | lines 6-12

A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
• The process could issue an I/O request and then it would be placed in an I/O queue.
• The process could create new sub process and will wait for its termination.
• The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Operating System module | PDF page 20 | lines 15-21

It is also called CPU scheduler. Main objective is increasing system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects process among the processes that are ready to execute and allocates CPU to one of them.
Short term scheduler also known as dispatcher, execute most frequently and makes the fine-grained decision of which process to execute next. Short term scheduler is faster than long term scheduler.

Question sources

Mock exam 1 · Question 22

Question 102 | Q01507

Under preemptive priority scheduling, which event sends a currently running, unfinished process back to the ready state?

A. Arrival of a higher-priority ready process that preempts it

B. Completion of its final CPU burst

C. Suspension while waiting for an event

D. Issuing a blocking I/O request

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer A: Arrival of a higher-priority ready process that preempts it

Preemption removes the CPU from a running process that can still execute, returning it to ready. In the stated priority policy, arrival of a higher-priority ready process triggers that handoff. Final completion ends execution, while a blocking I/O request makes the process wait for an event. Suspension while waiting also does not describe ordinary readiness, so only the explicit preemption event fits.

Why each choice is right or wrong

A - Correct: The higher-priority arrival preempts the runnable process and returns it to ready.

B - Incorrect: Final completion transitions to terminated, not ready.

C - Incorrect: Suspended waiting work is not eligible for the ordinary ready queue.

D - Incorrect: Blocking I/O transitions running to waiting.

Primary note proof

Study-note section OS06

> New processes are created/admitted; ready means eligible for CPU but not currently running. Running uses a CPU. Waiting/blocked means unable to proceed until an event; terminated means execution has ended. Running→ready follows time-slice expiry or another preemption. Running→waiting follows blocking I/O. I/O completion normally gives waiting→ready, not a guaranteed immediate running state.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job. Check whether a smaller or larger numeric value means higher priority. State whether a high-priority arrival preempts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 19 | lines 6-12

> A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
> • The process could issue an I/O request and then it would be placed in an I/O queue.
> • The process could create new sub process and will wait for its termination.
> • The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Operating System module | PDF page 24 | lines 10-15

> 7.3 Priority Based Scheduling
> • Each process is assigned a priority. Process with highest priority is to be executed first and so on.
> • Processes with same priority are executed on first come first serve basis.
> • Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 16 | lines 29-39

1. New The process is being created.
2.
Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
3.
Running Process instructions are being executed (i.e. The process that is currently being executed).
4.
Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
5. Terminated The process has finished execution.

Question sources

Mock exam 1 · Question 23

Question 103 | Q01508

One of the following deadlock handling methods is used if it occurs rarely and is insignificant.

A. Prevention

B. waiting

C. recovery

D. Ignorance

Hint

Separate handling before allocation from actions taken after detection, and consider the cost of checking.

Answer D: Ignorance, the intentional ignoring/ostrich policy.

The ostrich or ignoring policy accepts the risk of a sufficiently rare, insignificant deadlock problem when systematic handling is not worthwhile. That is the intended meaning of Ignorance. Prevention instead constrains requests to break a necessary deadlock condition, and recovery acts after detection. Merely waiting is not the same as deliberately choosing to tolerate the rare problem.

Why each choice is right or wrong

A - Incorrect: Prevention constrains resource requests to remove a necessary deadlock condition; it does not deliberately ignore the problem.

B - Incorrect: Waiting describes a process behavior, not the deliberate policy of accepting rare deadlocks.

C - Incorrect: Recovery takes action after deadlock detection rather than choosing to ignore the rare problem.

D - Correct: Ignorance names the intentional ignoring or ostrich policy in this context.

Primary note proof

Study-note section OS26

> Ignoring a sufficiently rare problem is sometimes called the ostrich policy, but real systems still prevent or detect selected deadlocks. “Deadlock acceptance” can mean intentional ignoring, not necessarily an invalid approach.

Study-note section OS24

> Prevention constrains how requests are made. Share a resource where safe to remove mutual exclusion; request all needed resources before holding any to remove hold-and-wait; permit safe resource withdrawal/restart to attack no-preemption; impose a global acquisition order to remove circular wait.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Mock exam 1 · Question 24

Question 104 | Q01587

One of the following is false about linked allocation.

A. Is based on sequential file access

B. Use pointer concept

C. No external fragmentation

D. The size of the file should be set in advanced

Hint

Trace how logical file blocks reach physical blocks and account for pointer metadata.

Answer D: The size of the file should be set in advanced

PDF p49 says linked allocation carries links to disk blocks, is effective for sequential access and has no external fragmentation. No advance contiguous size reservation is part of that linked structure; D confuses it with a fixed extent requirement.

Why each choice is right or wrong

A - Incorrect: Sequential access effectiveness is explicitly stated on p49.

B - Incorrect: Links/pointers are explicitly stated on p49.

C - Incorrect: No external fragmentation is explicitly stated on p49.

D - Correct: A chain can link another block without requiring one pre-sized contiguous extent; no advance-size requirement follows from the p49 linked structure.

Primary note proof

Study-note section OS38

> Contiguous allocation uses consecutive blocks; logical block i is start+i and supports direct access, but growth and external fragmentation are problems. Extents allow several contiguous runs. Linked allocation chains blocks and supports growth without reserving a complete contiguous file; locating block i requires traversing preceding links. FAT stores chains in a table; caching that table does not change the chain into a direct arithmetic mapping.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 49 | lines 30-35

> 13.3.2 Linked Allocation
> Each file carries a list of links to disk blocks.
> Directory contains link / pointer to first block of a file.
> No external fragmentation
> Effectively used in sequential access file.
> Inefficient in case of direct access file.

Operating System module | PDF page 48 | lines 5-8

> 13.1 File
> A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the file’s creator and user.

Question sources

Mock exam 3 · Question 19

Question 105 | Q01588

Consider three specified implementations: an Excel application process with several worker threads, a client-browser process with multiple threads sharing its address space, and a server process using only its single main thread. Which listed process does not use multithreading?

A. Excel application

B. Client browser

C. Server

D. None of the choices is correct

Hint

Compare per-thread execution context with shared process resources and the stated thread mapping.

Answer C: Server

Multithreading means that one process contains multiple threads of execution. The described spreadsheet and browser processes each have several threads sharing their process resources. The specified server implementation has only its main thread and therefore is not multithreaded. This conclusion follows from the supplied designs, not from a universal claim about every version of Excel, every browser, or every server.

Why each choice is right or wrong

A - Incorrect: The specified Excel process has several worker threads and is multithreaded.

B - Incorrect: The specified browser process has multiple threads and is multithreaded.

C - Correct: The specified server has only one thread and is not multithreaded.

D - Incorrect: C is a valid non-multithreaded implementation, so none is incorrect.

Primary note proof

Study-note section OS08

> A thread has its own PC, registers and stack. Threads in a process share code, address space and usually open-file resources. A process can contain many threads; a thread belongs to one process at a time. Threads are not a group of processes and need not execute sequentially.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 27 | lines 6-14

> A thread is a flow of execution through the process code, with its own program counter, system registers and stack. A thread is also called a light weight process. Threads provide a way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.
> Each thread belongs to exactly one process and no thread can exist outside a process. Each

thread represents a separate flow of control. Threads have been successfully used in implementing network servers and web server. They also provide a suitable foundation for parallel execution of applications on shared memory multiprocessors. Following figure shows

Question sources

Mock exam 3 · Question 20

Question 106 | Q01589

One of the following security techniques is not supported by modern operating systems.

A. Cryptography

B. Authentication

C. Firewall

D. None of the above

Hint

Distinguish identity checks, permission checks and the particular security property at issue.

Answer D: None: modern OS families can support all three techniques.

OS39 states that modern OS families can support cryptography, authentication and firewalls, while support/configuration depends on product. Therefore none of the three is categorically unsupported by modern operating systems. The answer is a capability-category statement, not a guarantee that every installation includes every feature or enables it by default. Each technique has a distinct security role.

Why each choice is right or wrong

A - Incorrect: Cryptography is explicitly a capability that modern OS families can support, though configuration varies.

B - Incorrect: Authentication is explicitly supported as a modern OS capability for verifying identity.

C - Incorrect: Firewalls are explicitly within modern OS support capabilities, even though product and configuration matter.

D - Correct: All three techniques can be supported; none is categorically outside modern operating-system security capabilities.

Primary note proof

Study-note section OS39

Password hashing verifies secrets without storing plaintext; one-time passwords expire after use. Encryption protects confidentiality under correct key use but does not automatically provide integrity. Trojans masquerade as useful software, worms propagate independently and viruses

attach to hosts; denial of service harms availability. Modern OS families can support cryptography, authentication and firewalls, but support/configuration depends on product.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 50 | lines 18-28

Authentication refers to identifying each user of the system and associating the executing programs with those users. It is the responsibility of the Operating System to create a protection system which ensures that a user who is running a particular program is authentic. Operating Systems generally identifies/authenticates users using following three ways:
Username / Password - User need to enter a registered username and password with Operating system to login into the system.
User card/key - User need to punch card in card slot, or enter key generated by key generator in option provided by operating system to login into the system.
User attribute - fingerprint/ eye retina pattern/ signature - User need to pass his/her attribute via designated input device used by operating system to login into the system.

Operating System module | PDF page 54 | lines 5-17

Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
Shell - Linux provides a special interpreter program which can be used to execute

Operating System module | PDF page 54 | lines 5-21

Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.

Shell - Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs etc.
Security - Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

Question sources

Mock exam 3 · Question 21

Question 107 | Q01627

Under the basic fixed-page and variable-length-segment memory models, which listed differences are correct? Select all that apply.

A. Paging leads to internal fragmentation whereas segmentation leads external fragmentation

B. The size of block is fixed in the case of paging but variable in the case of segmentation

C. Paging use page table whereas segmentation use segmentation table

D. Paging is visible to user whereas segmentation is invisible to user

Hint

Check the logical region and offset bound before computing its physical location.

Answer A, B, C: Paging leads to internal fragmentation whereas segmentation leads external fragmentation; The size of block is fixed in the case of paging but variable in the case of segmentation; Paging use page table whereas segmentation use segmentation table

Basic paging uses fixed-size pages and page tables, and its final allocated page can contain unused space. Basic segmentation uses variable-length logical units and segment tables, so free holes can cause external fragmentation. Those properties make A, B and C valid contrasts. D reverses the usual abstraction relationship: segments reflect program structure, while ordinary page placement is largely transparent to the application.

Why each choice is right or wrong

A - Correct: Paging can waste space internally; variable-length segmentation can fragment externally.

B - Correct: Pages have a fixed size while segments have variable lengths.

C - Correct: Paging uses page tables and segmentation uses segment tables.

D - Incorrect: Segments reflect program structure; ordinary pages are largely transparent, reversing this option.

Primary note proof

Study-note section OS31

Under the basic models, paging uses fixed units/page tables and can waste space internally; segmentation uses variable logical units/segment tables and can fragment externally. Segments reflect program structure whereas ordinary pages are largely transparent to applications. Apply all three contrasts when comparing paging and segmentation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 34 | lines 47-56

External fragmentation is avoided by using paging technique. Paging is a technique in which physical memory is broken into blocks of the same size called pages (size is power of 2, between 512 bytes and 8192 bytes). When a process is to be executed, it's corresponding pages are loaded into any available memory frames.
Logical address space of a process can be non-contiguous and a process is allocated physical memory whenever the free memory frame is available. Operating system keeps track of all free frames. Operating system needs n free frames to run a program of size n pages.
Address generated by CPU is divided into
• Page number (p) -- page number is used as an index into a page table which contains base address of each page in physical memory.

Operating System module | PDF page 34 | lines 34-45

External
fragmentation
Total memory space is enough to satisfy a request or to reside a
process in it, but it is not contiguous so it cannot be used.
2
Internal
fragmentation
Memory block assigned to process is bigger. Some portion of memory
is left unused as it cannot be used by another process.
External fragmentation can be reduced by compaction or shuffle memory contents to place all free memory together in one large block. To make compaction feasible, relocation should be dynamic

Operating System module | PDF page 35 | lines 7-12

Segmentation is a technique to break memory into logical pieces where each piece represents a group of related information. For example, data segments or code segment for each process, data segment for operating system and so on. Segmentation can be implemented using or without using paging.
Unlike paging, segment is having varying sizes and thus eliminates internal fragmentation. External fragmentation still exists but to lesser extent.

Operating System module | PDF page 36 | lines 3-7

Address generated by CPU is divided into
• Segment number (s) -- segment number is used as an index into a segment table which contains base address of each segment in physical memory and a limit of segment.

• Segment offset (o) -- segment offset is first checked against limit and then is combined with base address to define the physical memory address.

Question sources

Mock exam 2 · Question 16

Question 108 | Q01628

One of the following IO services is used store data in memory while transferring between devices.

A. Buffering

B. Scheduling

C. Caching

D. Spooling

Hint

Identify whether data is in transit, retained for reuse or queued for a device.

Answer A: Buffering

PDF p47 defines buffering as retaining data while transferred between devices or an application and device. This is A; caching stores reusable copies and spooling queues a noninterleavable output stream.

Why each choice is right or wrong

A - Correct: Directly matches transfer-time storage (p47).

B - Incorrect: Scheduling reorders requests, rather than providing transfer storage.

C - Incorrect: Caching keeps copies for more efficient access, not specifically transfer mismatch handling.

D - Incorrect: Spooling holds queued device output, a specialized buffer use; no printer/output serialization is specified in the stem.

Primary note proof

Study-note section OS35

Buffering temporarily holds data in transit to absorb speed/transfer-size differences. Caching retains copies to accelerate reuse. Spooling queues jobs for a device such as a printer, usually on disk, so producers need not own it throughout job creation. These have different purposes even if one implementation combines them.

Study-note section OS06

Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 47 | lines 5-9

> Buffering - Kernel I/O Subsystem maintains a memory area known as buffer that stores data while they are transferred between two devices or between a devices with an application operation. Buffering is done to cope with a speed mismatch between the producer and consumer of a data stream or to adapt between devices that have different data transfer sizes.

Operating System module | PDF page 47 | lines 10-11

> Caching - Kernel maintains cache memory which is region of fast memory that holds copies of data. Access to the cached copy is more efficient than access to the original.

Operating System module | PDF page 47 | lines 12-16

> Spooling and Device Reservation - A spool is a buffer that holds output for a device, such as a printer, that cannot accept interleaved data streams. The spooling system copies the queued spool files to the printer one at a time. In some operating systems, spooling is managed by a system daemon process. In other operating systems, it is handled by an in kernel thread.

Question sources

Mock exam 2 · Question 17

Question 109 | Q01629

Which directory structure describes the commonly presented namespace of modern general-purpose operating systems, with nested directories and absolute or relative paths? Choose the structural category, not every simpler arrangement that might also be implemented.

A. multi-level

B. All choices are correct

C. two level

D. Single level

Hint

Follow namespace structure and distinguish names, underlying files and mounted filesystems.

Answer A: multi-level

A multilevel or tree namespace supports nested directories and paths that can be absolute or relative. That is the structure the notes identify as commonly presented by modern general-purpose systems. A two-level structure separates users but does not express arbitrary nesting, and a single-level structure has only one namespace level. The aggregate therefore includes structures that do not match the requested hierarchical category.

Why each choice is right or wrong

A - Correct: Multilevel describes the hierarchical nested namespace.

B - Incorrect: Single-level and two-level arrangements do not both describe the requested nested tree, so all is incorrect.

C - Incorrect: Two-level separates users without providing the general nested hierarchy asked for.

D - Incorrect: Single-level has no nested directory hierarchy.

Primary note proof

Study-note section OS37

> A single-level directory requires unique names in one namespace. Two-level directories separate users. A tree allows nested directories and absolute/relative paths; links can create shared graphs. Modern general-purpose systems commonly present hierarchical namespaces, but a feature name alone does not prove that every system excludes simpler arrangements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 19-23

> A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Operating System does the following activities for file management.
> Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.

Operating System module | PDF page 54 | lines 5-17

> Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
> Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
> Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
> Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
> Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
> Shell - Linux provides a special interpreter program which can be used to execute

Question sources

Mock exam 2 · Question 18

Question 110 | SM0001

For the first four generations described in NYU's OS-history notes, which pairings incorrectly identify the defining historical characteristic of the named generation? Judge characteristic era, not whether a feature could also exist later. Select all that apply.

A. Second generation -> multiprogramming

B. Fourth generation -> real time systems

C. First generation → manual single-user operation without a separate resident OS

D. Third generation → uniprogrammed batch systems

Hint

Compare the stated historical clues and distinguish origin, characteristic era and technical capability.

Answer A, B, D: Second generation -> multiprogramming; Fourth generation -> real time systems; Third generation → uniprogrammed batch systems

The first four generations in NYU's teaching taxonomy associates the first era with one user and no separate OS, the second with uniprogrammed batch systems, the third with IC-based multiprogramming, and the fourth with personal computers. Thus A, B and D mismatch the characteristic eras, while C fits. OS02 supplies the operating-mode background; the generation associations come from the supplementary university history notes rather than the unchanged master note.

Why each choice is right or wrong

A - Correct: Multiprogramming characterizes the third era, so pairing it with the second is wrong.

B - Correct: The fourth-era heading is personal computers, not the introduction of real-time systems.

C - Incorrect: Manual single-user operation without a separate OS matches the first era.

D - Correct: Uniprogrammed batch characterizes the second era, not the defining third-era advance.

Primary note proof

Study-note section OS-S01

> In the first four generations of the cited NYU teaching taxonomy, the first generation uses vacuum tubes and manual single-user operation without a separate resident OS. The second generation pairs transistors with uniprogrammed batch systems; the resident OS starts successive jobs without being reloaded for every job.

Study-note section OS-S01

> The third generation pairs integrated circuits with multiprogramming. The fourth emphasizes personal computers. These are characteristic-era associations, not claims that each feature existed exclusively in one era.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 7 | lines 20-33

> Time sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multi-programmed Batch Systems and Time-Sharing Systems is that in case of multi-programmed batch systems, objective is to maximize processor use, whereas in Time-Sharing Systems objective is to minimize response time.
> Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, processor execute each user program in a short burst or quantum of computation. That is if n users are present, each user can get time quantum. When the user submits the command, the response time is in few seconds at most.
> Operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems

Operating System module | PDF page 13 | lines 21-25

> 4.3 Multiprogramming
> When two or more programs are residing in memory at the same time, then sharing the processor is referred to the multiprogramming. Multiprogramming assumes a single shared processor. Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has one to execute.

Operating System module | PDF page 7 | lines 5-7

> operating systems are there from the very first computer generation. Operating systems keep evolving over the period of time. Following are few of the important types of operating system which are most commonly used.

Question sources

MOE Computer Science Exit Exam · Question 1-1

National Computer Science Exit Exam · Question 16

National Computer Science Exit Exam · Question 27

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 211

Question 111 | SM0002

Which one of the following is asynchronous in message communication?

A. Non-block receive

B. Blocking receive

C. Blocking send

D. Direct message

Hint

Identify how data moves, which participant waits and what event allows the operation to continue.

Answer A: Nonblocking receive.

OS09 defines nonblocking operations as returning promptly with data, progress or a would-block result. Among the options, nonblocking receive therefore supplies the asynchronous/non-waiting behavior being contrasted. Blocking receive waits for a message, blocking send waits for its defined completion condition, and direct message concerns how endpoints are addressed rather than whether the call blocks.

Why each choice is right or wrong

A - Correct: Nonblocking receive returns promptly instead of waiting until a message arrives, matching this asynchronous contrast.

B - Incorrect: Blocking receive waits for a message rather than providing the non-waiting behavior requested.

C - Incorrect: Blocking send waits for its defined handoff or receipt condition rather than returning promptly.

D - Incorrect: Direct messaging describes endpoint addressing, not whether sending or receiving blocks.

Primary note proof

Study-note section OS09

> Blocking send waits for the required handoff/buffering/receipt condition defined by its API. Blocking receive waits for a message. Nonblocking operations return promptly with data, progress or a would-block result. “Sender blocked until received” identifies blocking send under rendezvous semantics.

Study-note section OS-S04

> Direct message communication names the receiving process explicitly. Indirect communication uses a mailbox or port. This naming distinction is independent of whether a send or receive blocks: naming a destination does not determine synchronization semantics.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 11 | lines 15-19

• Two processes often require data to be transferred between them.
• The both processes can be on the one computer or on different computer but are connected through computer network.
• Communication may be implemented by two methods either by Shared Memory or by Message Passing.

Question sources

MOE Computer Science Exit Exam · Question 1-2

National Computer Science Exit Exam · Question 3

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 212

Question 112 | SM0011

Which one of the following is NOT correct about threads?

A. Threads are the entities scheduled for execution on the CPU

B. It is a group of processes

C. Threads have program counters

D. Threads have registers to hold their working memory

Hint

Compare per-thread execution context with shared process resources and the stated thread mapping.

Answer B: It is a group of processes

PDF p27 defines a thread as a flow within one process, with its own PC and registers; p29 says kernel scheduling occurs on a thread basis. B incorrectly inverts containment by calling a thread a group of processes.

Why each choice is right or wrong

A - Incorrect: Thread-basis scheduling is explicit on p29.

B - Correct: Each thread belongs to exactly one process, not a collection of processes (p27).

C - Incorrect: Own program counter is explicit (p27).

D - Incorrect: Own system registers are explicit (p27); working memory should mean saved execution values rather than all shared process memory.

Primary note proof

Study-note section OS08

A thread has its own PC, registers and stack. Threads in a process share code, address space and usually open-file resources. A process can contain many threads; a thread belongs to one process at a time. Threads are not a group of processes and need not execute sequentially. Independent processes can exist; cooperating ones communicate/share data.

Study-note section OS08

User-level threads are managed by a library; kernel threads are known to the OS. In many-to-one mapping, a blocking kernel call can block all user threads and one kernel execution context cannot use multiple CPUs simultaneously. One-to-one permits another kernel thread to run when one blocks. Many-to-many multiplexes user threads over several kernel threads.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 27 | lines 6-14

A thread is a flow of execution through the process code, with its own program counter, system registers and stack. A thread is also called a light weight process. Threads provide a way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.
Each thread belongs to exactly one process and no thread can exist outside a process. Each thread represents a separate flow of control. Threads have been successfully used in implementing network servers and web server. They also provide a suitable foundation for parallel execution of applications on shared memory multiprocessors. Following figure shows

Operating System module | PDF page 29 | lines 4-16

In this case, thread management done by the Kernel. There is no thread management code in the application area. Kernel threads are supported directly by the operating system. Any application can be programmed to be multithreaded. All of the threads within an application are supported within a single process.
The Kernel maintains context information for the process as a whole and for individuals' threads within the process. Scheduling by the Kernel is done on a thread basis. The Kernel performs thread creation, scheduling and management in Kernel space. Kernel threads are generally slower to create and manage than the user threads.
Advantages
• Kernel can simultaneously schedule multiple threads from the same process on multiple processes.
• If one thread in a process is blocked, the Kernel can schedule another thread of the same process.

Question sources

MOE Computer Science Exit Exam · Question 3-3

National Computer Science Exit Exam · Question 84

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 213

Question 113 | SM0029

A computer expert would like to know the number of processes that are completed per hour to assess the performance of a machine. Which one of the following best describes the expert's assessment?

A. Waiting time

B. System throughput

C. Turnaround time

D. Response time

Hint

Construct a timeline and distinguish completion rates from individual elapsed-time measures.

Answer B: System throughput.

OS13 defines throughput as completed jobs divided by elapsed time. Counting process completions per hour is exactly that rate. Waiting, turnaround and response are delays associated with an individual process or an average of such delays; they are not completion counts per hour. The stated measurement therefore identifies system throughput without any scheduling trace.

Why each choice is right or wrong

A - Incorrect: Waiting time measures ready-queue delay, not the number of processes completed in an hour.

B - Correct: Completed processes divided by hours elapsed is system throughput here’s definition.

C - Incorrect: Turnaround measures arrival-to-completion duration for a process rather than a system completion rate.

D - Incorrect: Response measures arrival-to-first-start delay, not the number of jobs finished per unit time.

Primary note proof

Study-note section OS13

> For arrival A, first start S and completion C, response=S−A and turnaround=C−A. For a single CPU burst B with no I/O, waiting=C−A−B. With I/O, subtract all CPU and blocked time to isolate ready-queue waiting. Throughput=completed jobs/elapsed time; CPU utilization=busy time/elapsed time. Minimize response/turnaround/waiting under the selected goal and maximize useful throughput/utilization.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 31-35

Control over system performance -- Recording delays between request for a service and response from the system.
Job accounting -- Keeping track of time and resources used by various jobs and users.
Error detecting aids -- Production of dumps, traces, error messages and other debugging and error detecting aids.

Operating System module | PDF page 37 | lines 6-11

Virtual memory is a technique that allows the execution of processes which are not completely available in memory. The main visible advantage of this scheme is that programs can be larger than physical memory. Virtual memory is the separation of user logical memory from physical memory.
This separation allows an extremely large virtual memory to be provided for programmers when only a smaller physical memory is available. Following are the situations, when entire

Operating System module | PDF page 37 | lines 20-26

• Less number of I/O would be needed to load or swap each user program into memory.
• A program would no longer be constrained by the amount of physical memory that is available.
• Each user program could take less physical memory, more programs could be run the same time, with a corresponding increase in CPU utilization and throughput.
Virtual memory is commonly implemented by demand paging. It can also be implemented in a segmentation system. Demand segmentation can also be used to provide virtual memory.

Question sources

MOE Computer Science Exit Exam · Question 9-1

National Computer Science Exit Exam · Question 89

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 214

Question 114 | SM0075

A process is terminated by the operating system due to a reference to non-existing memory. Which one of the following best explains the process termination?

A. Normal exit

B. Error exit

C. Terminated by another process

D. Fatal exit

Hint

Separate creation, image replacement and termination; classify termination by its cause.

Answer D: Fatal exit.

OS07 explicitly identifies a reference to nonexistent/protected memory as a fatal fault. Here the OS terminates the process because of that fault, so this is involuntary fatal termination. It is not the program's voluntary error exit, normal completion or a kill request by another ordinary process. The cause in the stem resolves the otherwise loose exit terminology.

Why each choice is right or wrong

A - Incorrect: Normal exit is voluntary completion, unlike OS-enforced termination for an invalid memory reference.

B - Incorrect: A program-detected error exit is voluntary; the stem specifies an unrecoverable memory-access fault.

C - Incorrect: The stated cause is a fault handled by the OS, not a kill request from another process.

D - Correct: The note explicitly classifies nonexistent-memory access as a fatal fault causing involuntary termination.

Primary note proof

Study-note section OS07

> Normal completion and a program-detected error exit are voluntary; an unrecoverable fault or being killed is involuntary. exit() terminates normally through the runtime; abort() requests abnormal termination. A reference to nonexistent/protected memory is a fatal fault, unlike an application deciding to exit after bad input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 39 | lines 5-11

> Check an internal table for this process, to determine whether the reference was a valid or it was an invalid memory access.
> Step 2
> If the reference was invalid, terminate the process. If it was valid, but page have not yet brought in, page in the latter.
> Step 3 Find a free frame.
> Step 4 Schedule a disk operation to read the desired page into the newly allocated frame.

Operating System module | PDF page 16 | lines 29-39

> 1. New The process is being created.
> 2.
> Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
> 3.
> Running Process instructions are being executed (i.e. The process that is currently being executed).
> 4.

Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
5. Terminated The process has finished execution.

Question sources

MOE Computer Science Exit Exam · Question 21-3

National Computer Science Exit Exam · Question 1

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 216

Question 115 | SM0207

What is a characteristic feature of round-robin scheduling?

A. It assigns priorities to tasks based on their relative importance

B. It always selects the task with the shortest remaining time

C. It favors short tasks over long tasks

D. It guarantees fairness by providing each task with an equal time slice

Hint

Trace bounded turns in ready-queue order and include the stated overhead and arrivals.

Answer D: It guarantees fairness by providing each task with an equal time slice

PDF p25 gives each process a fixed quantum and preempts after it. D identifies that equal-turn mechanism; it guarantees equal-sized opportunities, not identical lifetime CPU totals for unequal jobs.

Why each choice is right or wrong

A - Incorrect: Priority rank is the p24 policy, not the RR criterion.

B - Incorrect: Remaining-time selection is not RR's fixed quantum.

C - Incorrect: Short-job preference is not RR's fixed quantum rule.

D - Correct: Equal quantum per turn is precisely the p25 definition.

Primary note proof

Study-note section OS18

RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Study-note section OS17

Priority scheduling chooses the highest-priority ready job.

Study-note section OS16

Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs. SRTF reconsiders on arrivals and selects the shortest remaining work; a shorter newcomer can preempt.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 2-3

Computer Science Exit Exam practice collection · Question 226

Question 116 | SM0220

Which objective of modern operating systems involves providing a consistent and convenient environment for running applications?

A. Maximizing hardware utilization

B. Ensuring fault tolerance

C. Providing security and protection

D. Providing a user-friendly interface

Hint

Separate broad resource-management goals from individual services or applications.

Answer D: Providing a user-friendly interface

PDF p5 emphasizes a convenient program-execution environment; p14 supplies a user interface with short interactive response. D is the option specifically about user convenience.

Why each choice is right or wrong

A - Incorrect: Utilization concerns efficiency rather than the stated convenient interaction goal.

B - Incorrect: Fault tolerance concerns failure behavior, not ordinary interaction convenience.

C - Incorrect: Security controls authorization (p6), not the named convenience goal.

D - Correct: Interface support is the module's interaction/convenience role (pp5,14).

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process. Application development tools use OS services but application development itself is not a core resource-management function.

Study-note section OS01

> Updates and monitoring help administration, but automatic updates are not a defining property of every OS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 5 | lines 6-13

> An operating System (OS) is an intermediary between users and computer hardware. It provides users an environment in which a user can execute programs conveniently and efficiently. In technical terms, it is software which manages hardware. An operating System controls the allocation of resources and services such as memory, processors, devices and information.
> 1.1 Definition
> An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Operating System module | PDF page 14 | lines 18-23

> Interactivity refers that a User is capable to interact with computer system. Operating system does the following activities related to interactivity.
> • OS provides user an interface to interact with system.
> • OS managers input devices to take inputs from the user. For example, keyboard.

• OS manages output devices to show outputs to the user. For example, Monitor.
• OS Response time needs to be short since the user submits and waits for the result.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 7-1

Computer Science Exit Exam practice collection · Question 224

Question 117 | SM0228

All of the following are scheduling criteria EXCEPT:

A. CPU utilization

B. Min Turnaround Time

C. Max Throughput

D. Max Response Time

Hint

Construct a timeline and distinguish completion rates from individual elapsed-time measures.

Answer D: Maximizing response time is the wrong optimization direction.

OS13 says to minimize response, turnaround and waiting while maximizing useful throughput/utilization under the chosen objective. Thus maximum response time is the mismatched criterion: making first response later worsens responsiveness. The other alternatives name a standard metric or its conventional optimization direction. The question is about preferred direction, not whether response time itself is measurable.

Why each choice is right or wrong

A - Incorrect: CPU utilization is a standard scheduling metric measuring the fraction of elapsed time the CPU is busy.

B - Incorrect: Minimizing turnaround is the conventional goal for reducing arrival-to-completion delay.

C - Incorrect: Maximizing useful throughput is a conventional goal for increasing completed work per unit time.

D - Correct: Response delay should ordinarily be minimized; maximizing it reverses the responsiveness objective.

Primary note proof

Study-note section OS13

For arrival A, first start S and completion C, response=S−A and turnaround=C−A. For a single CPU burst B with no I/O, waiting=C−A−B. With I/O, subtract all CPU and blocked time to isolate ready-queue waiting. Throughput=completed jobs/elapsed time; CPU utilization=busy

time/elapsed time. Minimize response/turnaround/waiting under the selected goal and maximize useful throughput/utilization.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 7 | lines 20-33

Time sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multi-programmed Batch Systems and Time-Sharing Systems is that in case of multi-programmed batch systems, objective is to maximize processor use, whereas in Time-Sharing Systems objective is to minimize response time.
Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, processor execute each user program in a short burst or quantum of computation. That is if n users are present, each user can get time quantum. When the user submits the command, the response time is in few seconds at most.
Operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems

Operating System module | PDF page 13 | lines 21-25

4.3 Multiprogramming
When two or more programs are residing in memory at the same time, then sharing the processor is referred to the multiprogramming. Multiprogramming assumes a single shared processor. Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has one to execute.

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 37 | lines 20-26

• Less number of I/O would be needed to load or swap each user program into memory.
• A program would no longer be constrained by the amount of physical memory that is available.
• Each user program could take less physical memory, more programs could be run the same time, with a corresponding increase in CPU utilization and throughput.
Virtual memory is commonly implemented by demand paging. It can also be implemented in a segmentation system. Demand segmentation can also be used to provide virtual memory.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 9-3

Computer Science Exit Exam practice collection · Question 225

Question 118 | SM0241

Which umbrella function of modern operating systems allocates and coordinates hardware resources such as CPU time, memory and I/O devices?

A. File Management

B. Resource Management

C. Process Management

D. Security Management

Hint

Separate broad resource-management goals from individual services or applications.

Answer B: Resource Management

An OS allocates CPU time, memory, storage, devices and communication resources. The umbrella term for coordinating these allocations is resource management. File, process and security management cover narrower responsibilities, so none alone names the full scope of the question. Replacing device management with the umbrella category preserves the combined CPU/memory/I/O concept and gives the item one complete answer.

Why each choice is right or wrong

A - Incorrect: File management concerns files and storage abstractions rather than all named resources.

B - Correct: Resource management spans CPU time, memory and I/O devices.

C - Incorrect: Process management covers execution and CPU scheduling but is narrower than the combined scope.

D - Incorrect: Security management controls protection and permissions rather than naming all allocation functions.

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs.

Study-note section OS06

> Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Study-note section OS36

A file has content and metadata such as name, size, owner, permissions and timestamps. open obtains an open-file reference, read/write transfer bytes, seek changes an offset, close releases a reference, truncate changes size and unlink removes a name.

Study-note section OS39

Authentication verifies identity; authorization checks permitted operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 4-9

In multiprogramming environment, OS decides which process gets the processor when and how much time. This function is called process scheduling. Operating System does the following activities for processor management. Keeps tracks of processor and status of process. Program responsible for this task is known as traffic controller.
Allocates the processor (CPU) to a process.
De-allocates processor when processor is no longer required.

Operating System module | PDF page 5 | lines 26-36

Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address. Main memory provides a fast storage that can be access directly by the CPU. So, for a program to be executed, must in the main memory.
Activities Operating System for memory management.
Keeps tracks of primary memory (i.e. what part of it are in use by whom, what part are not in use.)
In multiprogramming, OS decides which process will get memory when and how much.
Allocates the memory when the process requests it to do so.
De-allocates the memory when the process no longer needs it or has been terminated.

Operating System module | PDF page 6 | lines 11-17

OS manages device communication via their respective drivers. Activities of Operating System for device management.
Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.
Decides which process gets the device when and for how much time.
Allocates the device in the efficient way.
De-allocates devices.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 13-3

Computer Science Exit Exam practice collection · Question 223

Question 119 | SM0268

Which scheduling algorithm is commonly associated with the concept of aging, where the priority of a task increases the longer it waits in the ready queue?

A. Round Robin (RR)

B. Shortest Job Next (SJN)

C. Priority Scheduling

D. Shortest Remaining Time First (SRTF)

Hint

Check priority direction, preemption and whether waiting can change effective rank.

Answer C: Priority scheduling.

OS17 describes aging as increasing waiting jobs' effective priority to reduce indefinite postponement. That is a direct adjustment within a priority scheme and matches the stem exactly. RR rotates turns, SJN selects the shortest next burst, and SRTF selects the shortest remaining work. Those policies may be modified, but they are not the named priority-aging mechanism requested.

Why each choice is right or wrong

A - Incorrect: RR cycles ready jobs by quantum rather than defining aging through increasing priority.

B - Incorrect: SJN chooses shortest next bursts; aging refers specifically to adjusting the effective priority of waiting work.

C - Correct: Priority scheduling can raise a waiting task's effective priority through aging to reduce starvation.

D - Incorrect: SRTF compares remaining bursts rather than increasing priority solely because a task has waited longer.

Primary note proof

Study-note section OS17

> Aging increases waiting jobs' effective priority according to a rule, reducing indefinite postponement when the priority scheme and workload permit.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn.

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs. SRTF reconsiders on arrivals and selects the shortest remaining work; a shorter newcomer can preempt.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 24 | lines 10-15

> 7.3 Priority Based Scheduling
> • Each process is assigned a priority. Process with highest priority is to be executed first and so on.
> • Processes with same priority are executed on first come first serve basis.
> • Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 22-2

Computer Science Exit Exam practice collection · Question 222

Question 120 | SM0278

In a system susceptible to deadlock, what does the 'no preemption' condition imply?

A. Resources can be preempted and assigned to other processes

B. Processes cannot request additional resources while holding others

C. Resources cannot be preempted while they are in use

D. Processes must release all resources when blocked

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer C: Allocated resources cannot be forcibly withdrawn.

OS22 defines no resource preemption as requiring voluntary release by the holder. C states that resources cannot be preempted while in use and therefore matches. A reverses the condition. B and D describe policies attacking hold-and-wait by limiting requests or requiring release; they are different from the no-preemption property itself.

Why each choice is right or wrong

A - Incorrect: Permitting withdrawal and reassignment is resource preemption, directly reversing the condition requested.

B - Incorrect: Forbidding additional requests while holding resources attacks hold-and-wait rather than defining no-preemption.

C - Correct: The holder retains the resource until voluntary release, so it cannot simply be forcibly withdrawn.

D - Incorrect: Requiring release when blocked removes holding-while-waiting; it is not the no-preemption definition.

Primary note proof

Study-note section OS22

> Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Study-note section OS24

> “If waiting, release all held resources and try again” removes simultaneous holding while waiting.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 25-3

Computer Science Exit Exam practice collection · Question 227

Question 121 | SS106

When a process is in a "Blocked" state waiting for some Input-Output services and afterthe service is completed, it will goes to a state known as:

A. Terminated

B. Suspended

C. Ready

D. Running

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer C: Ready

After I/O completion the process no longer waits for that event but must obtain CPU allocation, so it becomes Ready (PDF pp16,19-20). Running follows only when the scheduler actually selects it.

Why each choice is right or wrong

A - Incorrect: Completion of an I/O request is not completion of the process.

B - Incorrect: Suspension concerns memory removal/swapping, not an inevitable I/O-completion transition.

C - Correct: Ready means eligible and waiting for processor assignment (p16).

D - Incorrect: It can run after scheduling; I/O completion alone does not allocate CPU (p20).

Primary note proof

Study-note section OS06

> New processes are created/admitted; ready means eligible for CPU but not currently running. Running uses a CPU. Waiting/blocked means unable to proceed until an event; terminated means execution has ended. Running→ready follows time-slice expiry or another preemption. Running→waiting follows blocking I/O. I/O completion normally gives waiting→ready, not a guaranteed immediate running state.

Study-note section OS06

> Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 29-39

> 1. New The process is being created.
> 2.
> Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
> 3.
> Running Process instructions are being executed (i.e. The process that is currently being executed).
> 4.
> Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
> 5. Terminated The process has finished execution.

Operating System module | PDF page 19 | lines 6-12

> A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
> • The process could issue an I/O request and then it would be placed in an I/O queue.
> • The process could create new sub process and will wait for its termination.
> • The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Operating System module | PDF page 20 | lines 15-21

It is also called CPU scheduler. Main objective is increasing system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects process among the processes that are ready to execute and allocates CPU to one of them.
Short term scheduler also known as dispatcher, execute most frequently and makes the fine-grained decision of which process to execute next. Short term scheduler is faster than long term scheduler.

Question sources

Computer Science model exam collection · Question 13

Question 122 | SS107

A memory management policy that decisions must be made as to which page or pages are to be supplanted when a memory is full is known as:

A. Cleaning policy

B. Fetch policy

C. Load policy

D. Replacement policy

Hint

Trace every reference, distinguish hits from faults and update the policy state consistently.

Answer D: Replacement policy

Page replacement chooses which memory pages to swap out when allocation requires unavailable free space (PDF p40). D names that victim-selection policy.

Why each choice is right or wrong

A - Incorrect: Cleaning concerns writing dirty contents; it does not identify victim selection in the p40 definition.

B - Incorrect: Fetching concerns bringing data in, rather than selecting a resident victim.

C - Incorrect: Loading concerns admission/bringing in work, rather than selecting pages to evict.

D - Correct: Replacement is explicitly the decision of which pages to swap out (p40).

Primary note proof

Study-note section OS33

Replacement chooses a victim when no suitable free frame is available.

Study-note section OS-S03

A fetch policy decides when contents are brought into memory; demand paging fetches a valid nonresident page when it is accessed. A cleaning policy concerns writing modified contents to

backing storage. A replacement policy chooses which resident page to evict. These decisions can cooperate during one page fault, but they are different decisions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 40 | lines 7-20

Page replacement algorithms are the techniques using which Operating System decides which memory pages to swap out, write to disk when a page of memory needs to be allocated.
Paging happens whenever a page fault occurs and a free page cannot be used for allocation purpose accounting to reason that pages are not available or the number of free pages is lower than required pages.
When the page that was selected for replacement and was paged out, is referenced again then it has to read in from disk, and this requires for I/O completion. This process determines the quality of the page replacement algorithm: the lesser the time waiting for page-ins, the better is the algorithm. A page replacement algorithm looks at the limited information about accessing the pages provided by hardware, and tries to select which pages should be replaced to minimize the total number of page misses, while balancing it with the costs of primary storage and processor time of the algorithm itself. There are many different page replacement algorithms. We evaluate an algorithm by running it on a particular string of memory reference and computing the number of page faults.

Question sources

Computer Science model exam collection · Question 14

Question 123 | SS122

Which of the following is not a principal event that cause processes to be created?

A. Multiple execution of processes

B. Execution of a process-creation system call by a running process

C. System initialization

D. Auser request to create a new process

Hint

Separate creation, image replacement and termination; classify termination by its cause.

Answer A: Multiple execution is not itself a process-creation event.

OS07 lists initialization, user requests, service requests and a running process's creation call as creation causes. B, C and D match that list. Multiple execution of processes describes work being run, not an explicit request or initialization event creating a new process. Existing processes can execute repeatedly without creating a child, so A is the unsupported creation trigger.

Why each choice is right or wrong

A - Correct: Executing existing processes multiple times is not itself the explicit creation trigger identified here.

B - Incorrect: A running process's creation system call is explicitly one of the listed process-creation causes.

C - Incorrect: System initialization is explicitly listed as a time when processes are created.

D - Incorrect: A user request to create a process is explicitly a valid process-creation cause.

Primary note proof

Study-note section OS07

> Processes arise during initialization, user requests, service requests or a running process's creation call.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 19 | lines 6-12

> A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
> • The process could issue an I/O request and then it would be placed in an I/O queue.
> • The process could create new sub process and will wait for its termination.
> • The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Operating System module | PDF page 16 | lines 6-9

> A process is a program in execution. The execution of a process must progress in a sequential fashion. Definition of process is following.
> • A process is defined as an entity which represents the basic unit of work to be implemented in the system.

Question sources

Computer Science model exam collection · Question 29

Question 124 | SS125

Which statement about Banker's algorithm is false?

A. It is a CPU scheduling algorithm

B. It uses each process's declared maximum resource claims

C. Its safety test simulates completions to check whether every process can finish

D. It checks a received resource request before granting it

Hint

Compute remaining needs and simulate feasible completions before deciding whether an allocation is safe.

Answer A: It is a CPU scheduling algorithm

Banker's algorithm is a deadlock-avoidance technique, not a CPU scheduling policy. It requires maximum resource claims, computes remaining needs, and simulates possible completions to check for a safe sequence. A request is evaluated after receipt and before being granted, including checks against need and availability. The factual alternatives therefore describe the supported safety procedure, leaving only the CPU-scheduling classification false.

Why each choice is right or wrong

A - Correct: Banker is deadlock avoidance rather than CPU scheduling, so this statement is false.

B - Incorrect: Maximum resource claims are required input, so this statement is true.

C - Incorrect: The safety test repeatedly simulates feasible completions, so this statement is true.

D - Incorrect: A request is checked before granting it, so this statement is true.

Primary note proof

Study-note section OS25

> Avoidance allows requests only if the resulting state remains safe. Banker requires maximum claims. Compute Need=Max−Allocation. Set Work=Available and repeatedly find an unfinished process with Need≤Work componentwise; simulate completion by adding its Allocation to Work. Finishing all gives a safe sequence. Tentatively grant a request only if it is within Need and Available and the safety test succeeds. Unsafe does not mean already deadlocked. Banker is avoidance, not CPU scheduling, detection or prevention by breaking a condition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Computer Science model exam collection · Question 32

Question 125 | SS138

A set of processes is blocked: each holds a resource and waits for another resource that only another blocked member of the set can release. No member can proceed without intervention. This state is called:

A. Deadlock

B. Queuing

C. Preemption

D. Overloading

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer A: Deadlock

The specified set has a closed dependency: every member is waiting for something only another blocked member can provide. None can reach the release operation needed by another, so the state is deadlock. Queuing can occur during ordinary waits that eventually finish; preemption withdraws or reallocates a resource; overload concerns excessive demand. The added dependency condition makes the deadlock interpretation definite.

Why each choice is right or wrong

A - Correct: The closed dependency among blocked members defines deadlock.

B - Incorrect: Ordinary queuing does not require a dependency that prevents every member from progressing.

C - Incorrect: Preemption is resource withdrawal or reassignment rather than this blocked state.

D - Incorrect: Overload does not by itself create the specified closed dependency.

Primary note proof

Study-note section OS22

> A set is deadlocked when each member waits for an event/resource that only another blocked member can provide. Mere waiting for another holder is normal blocking; indefinite unfair delay can be starvation without a cycle of dependencies.

Study-note section OS24

> Prevention constrains how requests are made. Share a resource where safe to remove mutual exclusion; request all needed resources before holding any to remove hold-and-wait; permit safe resource withdrawal/restart to attack no-preemption; impose a global acquisition order to remove circular wait.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 29-39

1. New The process is being created.
2.
Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
3.
Running Process instructions are being executed (i.e. The process that is currently being executed).
4.
Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
5. Terminated The process has finished execution.

Operating System module | PDF page 19 | lines 6-12

A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
• The process could issue an I/O request and then it would be placed in an I/O queue.
• The process could create new sub process and will wait for its termination.
• The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Operating System module | PDF page 10 | lines 23-28

• Loads a program into memory.
• Executes the program.
• Handles program's execution.
• Provides a mechanism for process synchronization.
• Provides a mechanism for process communication.
• Provides a mechanism for deadlock handling.

Question sources

Computer Science model exam collection · Question 45

Question 126 | SS183

Identify the lowest layer of Android architecture.

A. Application

B. Application Framework

C. Database

D. Linux Kernel

Hint

Use the stated historical or platform scope and distinguish origin from technical capability.

Answer D: Linux kernel.

OS40 directly states that Android's base kernel layer is Linux and places framework and applications above it. Thus Linux Kernel is the lowest offered layer. A database is not identified as the kernel foundation in this model. The answer rests on Android's architecture, not on a separate claim about Android's current market share.

Why each choice is right or wrong

A - Incorrect: Applications sit above the framework and kernel rather than forming Android's base layer.

B - Incorrect: The application framework is explicitly a higher layer than the Linux kernel foundation.

C - Incorrect: A database is not the base kernel layer identified here's Android architecture description.

D - Correct: The note explicitly names Linux as Android's base kernel layer beneath framework and applications.

Primary note proof

Study-note section OS40

> Android's base kernel layer is Linux; its framework and applications are higher layers. AOSP is open source, while a shipped device can include proprietary applications/services.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 54 | lines 24-30

> Hardware layer - Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc.).
> Kernel - Core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.
> Shell - An interface to kernel, hiding complexity of kernel's functions from users. Takes commands from user and executes kernel's functions.
> Utilities - Utility programs giving user most of the functionalities of an operating systems.

Question sources

Computer Science model exam collection · Question 90

Question 127 | AM0654

Which objective specifically concerns efficient allocation of CPU time, memory, storage and devices among competing programs?

A. To provide an interface for users to interact with the hardware

B. To manage and allocate system resources efficiently

C. To ensure data security and protection

D. To optimize program execution speed

Hint

Separate broad resource-management goals from individual services or applications.

Answer B: To manage and allocate system resources efficiently

Resource management allocates CPU time, memory, storage, devices and communications to programs. That directly matches efficient management and allocation in B. User interfaces support convenience, protection controls permitted access, and execution speed is a performance concern. Those are related goals, but the stated question asks specifically for the allocation objective rather than ranking every legitimate OS goal as universally most important.

Why each choice is right or wrong

A - Incorrect: Providing an interface concerns convenience rather than the specific allocation objective.

B - Correct: Efficient management and allocation directly names the requested resource objective.

C - Incorrect: Security and protection concern access control rather than allocation efficiency itself.

D - Incorrect: Program speed is a performance concern and is not equivalent to efficient allocation of all resources.

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process. Application development tools use OS services but application development itself is not a core resource-management function.

Study-note section OS39

> Authentication verifies identity; authorization checks permitted operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 5 | lines 6-13

An operating System (OS) is an intermediary between users and computer hardware. It provides users an environment in which a user can execute programs conveniently and efficiently. In technical terms, it is software which manages hardware. An operating System controls the allocation of resources and services such as memory, processors, devices and information.
1.1 Definition
An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Question sources

Computer Science MCQ practice collection · Question 654

Question 128 | AM0655

Which of the following is not a function of an operating system?

A. Process management

B. File management

C. Network management

D. Application development

Hint

Separate broad resource-management goals from individual services or applications.

Answer D: Application development is not a core OS resource-management function.

OS01 explicitly distinguishes application-development tools using OS services from application development itself as a core OS function. Processes, files and communication resources are managed by the OS. Therefore D is the exclusion in the intended resource-management scope. Providing compilers, editors or development APIs does not mean the operating system itself performs application development as its defining resource function.

Why each choice is right or wrong

A - Incorrect: Managing execution/process resources belongs to the OS's core resource-management responsibilities.

B - Incorrect: File and storage management are core OS services rather than the requested non-function.

C - Incorrect: Communication/network resource management belongs to the OS responsibilities described in the note.

D - Correct: The note explicitly separates developing applications from the OS functions used by development tools.

Primary note proof

Study-note section OS01

An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process. Application development tools use OS services but application development itself is not a core resource-management function.

Study-note section OS04

Process control, file/device management, information retrieval and communication are service categories.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 4-9

In multiprogramming environment, OS decides which process gets the processor when and how much time. This function is called process scheduling. Operating System does the following activities for processor management. Keeps tracks of processor and status of process. Program responsible for this task is known as traffic controller.
Allocates the processor (CPU) to a process.
De-allocates processor when processor is no longer required.

Operating System module | PDF page 6 | lines 19-23

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Operating System does the following activities for file management.
Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.

Operating System module | PDF page 8 | lines 21-26

Network Operating System runs on a server and provides server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks. Examples of network operating systems are Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.

Operating System module | PDF page 10 | lines 5-15

An Operating System provides services to both the users and to the programs.
• It provides programs, an environment to execute.
• It provides users, services to execute the programs in a convenient manner.
Operating Systems can provide the following common services:
✓ Program execution
✓ I/O operations
✓ File System manipulation
✓ Communication
✓ Error Detection

✓ Resource Allocation
✓ Protection

Question sources

Computer Science MCQ practice collection · Question 655

Question 129 | AM0656

What is the purpose of memory management in an operating system?

A. To provide an interface for users to interact with the computer system

B. To manage the execution of multiple processes simultaneously

C. To allocate and deallocate memory for processes and data

D. To ensure data integrity and security

Hint

Read the cited conceptual rules, state the model, and apply them before comparing the alternatives.

Answer C: To allocate and deallocate memory for processes and data

PDF pp5 and 31 say memory management tracks primary memory, decides allocations, and updates status when memory is freed. C is the specific allocation/deallocation function.

Why each choice is right or wrong

A - Incorrect: User interaction is a separate interface function.

B - Incorrect: Multiple-process execution is process/scheduling work; memory management supports it but does not define execution control.

C - Correct: Allocation and deallocation are the direct memory-management tasks (pp5,31).

D - Incorrect: Protection is supported by base/limit registers, but data integrity/security is too broad to identify the principal allocation role.

Primary note proof

Study-note section OS-S03

Memory management tracks which primary-memory regions are free or in use, assigns memory to processes and data, and deallocates it when no longer needed or when a process terminates. Allocation and deallocation concern memory ownership; CPU selection belongs to scheduling.

Study-note section OS28

In the standard base/limit model, base is that process's physical starting address and limit is its permitted length. Check 0≤logical_offset<limit, then physical=base+offset.

Study-note section OS03

A shell reads/interprets user commands; a GUI is another interface.

Study-note section OS06

Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Study-note section OS39

Authentication verifies identity; authorization checks permitted operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 31 | lines 24-31

Memory management is the functionality of an operating system which handles or manages primary memory. Memory management keeps track of each and every memory location either it is allocated to some process or it is free. It checks how much memory is to be allocated to processes. It decides which process will get memory at what time. It tracks whenever some memory gets freed or unallocated and correspondingly it updates the status.
Memory management provides protection by using two registers, a base register and a limit register. The base register holds the smallest legal physical memory address and the limit register specifies the size of the range. For example, if the base register holds 300000 and the

Operating System module | PDF page 5 | lines 26-36

Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address. Main memory provides a fast storage that can be access directly by the CPU. So, for a program to be executed, must in the main memory.
Activities Operating System for memory management.
Keeps tracks of primary memory (i.e. what part of it are in use by whom, what part are not in use.)
In multiprogramming, OS decides which process will get memory when and how much.
Allocates the memory when the process requests it to do so.
De-allocates the memory when the process no longer needs it or has been terminated.

Question sources

Computer Science MCQ practice collection · Question 656

Question 130 | AM0657

Which OS component converts generic I/O requests into operations specific to a particular device, rather than naming the kernel as a whole?

A. Kernel

B. Scheduler

C. File system

D. Device drivers

Hint

Separate device hardware, device-specific software and general event or transfer mechanisms.

Answer D: Device drivers

A device driver translates OS I/O requests into the particular operations a device requires. The kernel provides broader resource management and can contain drivers, but the question asks for that specific hardware-facing component. The scheduler selects runnable work and the filesystem organizes file abstractions. Neither performs the general per-device request translation described, making device drivers the precise answer.

Why each choice is right or wrong

A - Incorrect: The kernel is the larger system containing or coordinating the specific component.

B - Incorrect: The scheduler chooses runnable CPU work rather than translating device commands.

C - Incorrect: The filesystem organizes files rather than translating requests for every device type.

D - Correct: Drivers convert OS requests into device-specific operations.

Primary note proof

Study-note section OS34

> A controller is device hardware exposing status, control and data registers. A driver converts OS requests into device-specific operations. Device independence and uniform naming hide brand-specific details; error handling reports/retries supported failures.

Study-note section OS03

> The kernel implements core resource management, protection and system-call handling.

Study-note section OS06

> Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 47 | lines 19-26

12.5 Device driver
Device driver is a program or routine developed for an I/O device. A device driver implements I/O operations or behaviors on a specific class of devices. For example a system supports one or a number of multiple brands of terminals, all slightly different terminals may have a single terminal driver. In the layered structure of I/O system, device driver lies between interrupt handler and device independent I/O software. The job of a device driver is following.
To accept request from the device independent software above it.
To see to it that the request is executed.

Operating System module | PDF page 45 | lines 28-33

Application I/O Interface represents the structuring techniques and interfaces for the operating system to enable I/O devices to be treated in a standard, uniform way. The actual differences lies kernel level modules called device drivers which are custom tailored to corresponding devices but show one of the standard interfaces to applications. The purpose of the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, such as the I/O system calls. Following are the characteristics of I/O

Operating System module | PDF page 47 | lines 5-9

Buffering - Kernel I/O Subsystem maintains a memory area known as buffer that stores data while they are transferred between two devices or between a devices with an application operation. Buffering is done to cope with a speed mismatch between the producer and consumer of a data stream or to adapt between devices that have different data transfer sizes.

Question sources

Computer Science MCQ practice collection · Question 657

Question 131 | AM0658

What is the role of the file system in an operating system?

A. To manage system resources and allocate them to processes

B. To provide a user interface for interacting with the computer system

C. To store and organize files on secondary storage devices

D. To schedule and prioritize the execution of processes

Hint

Distinguish a file operation, a resource handle, an offset and file metadata.

Answer C: To store and organize files on secondary storage devices

Files are related information on secondary storage (PDF p48); file management tracks locations and organizes directories (p6). C gives that role.

Why each choice is right or wrong

A - Incorrect: System-wide allocation is broader than organizing files.

B - Incorrect: A user interface is not the file system's stored-information role.

C - Correct: Matches secondary storage and organization (pp6,48).

D - Incorrect: Process execution ordering belongs to schedulers (p20), not file organization.

Primary note proof

Study-note section OS36

> A file has content and metadata such as name, size, owner, permissions and timestamps. open obtains an open-file reference, read/write transfer bytes, seek changes an offset, close releases a reference, truncate changes size and unlink removes a name.

Study-note section OS37

> A single-level directory requires unique names in one namespace. Two-level directories separate users. A tree allows nested directories and absolute/relative paths; links can create shared graphs. Modern general-purpose systems commonly present hierarchical namespaces, but a feature name alone does not prove that every system excludes simpler arrangements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 48 | lines 5-8

> 13.1 File
> A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the file's creator and user.

Operating System module | PDF page 6 | lines 19-23

> A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Operating System does the following activities for file management.
> Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.

Question sources

Computer Science MCQ practice collection · Question 658

Question 132 | AM0659

Which scheduling algorithm is based on assigning a fixed time slice to each process in a round-robin manner?

A. First-Come, First-Served (FCFS) scheduling

B. Shortest Job Next (SJN) scheduling

C. Priority scheduling

D. Round-Robin scheduling

Hint

Trace bounded turns in ready-queue order and include the stated overhead and arrivals.

Answer D: Round-Robin scheduling

PDF p25 explicitly defines a fixed execution quantum followed by preemption and another process's turn. D is that round-robin policy.

Why each choice is right or wrong

A - Incorrect: FCFS uses arrival order without this specified repeating quantum (p23).

B - Incorrect: SJN selects short jobs using known execution length (p23).

C - Incorrect: Priority uses assigned rank (p24).

D - Correct: Fixed quantum plus repeated turns is the p25 RR definition.

Primary note proof

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.
> • Easy to understand and implement.
> • Poor in performance as average wait time is high.
> Wait time of each process is following
> Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science MCQ practice collection · Question 659

Question 133 | AM0660

What is the primary goal of file-access authorization in an operating system?

A. To ensure the availability of files for authorized users

B. To prevent unauthorized access to files and data

C. To optimize file storage and retrieval performance

D. To minimize the impact of system failures on file operations

Hint

Distinguish identity checks, permission checks and the particular security property at issue.

Answer B: To prevent unauthorized access to files and data

Authorization checks whether a user or other principal is permitted to perform an operation on a file. Preventing unauthorized access is therefore the direct goal. Availability and recovery protect access through failures or disruption, while storage optimization concerns performance. Those are useful objectives, but they are distinct from deciding who may read, write or otherwise access a particular file.

Why each choice is right or wrong

A - Incorrect: Availability keeps files accessible when needed; it is distinct from deciding whether an attempted access is authorized.

B - Correct: Preventing unauthorized access directly follows from checking permitted operations and file rights.

C - Incorrect: Storage and retrieval performance do not determine a caller’s access rights.

D - Incorrect: Limiting failure impact concerns resilience and recovery, rather than the access-permission decision.

Primary note proof

Study-note section OS39

> Authentication verifies identity; authorization checks permitted operations. An ACL lists principals and rights; owner/group/other bits are a simpler policy model, not universal defaults granting all owners full access or everyone read. Read, write, execute, append and delete are different rights, with filesystem-specific rules.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 29-30

> Security -- By means of password and similar other techniques, preventing unauthorized access to programs and data.

Operating System module | PDF page 11 | lines 3-8

> • Program needs to read a file or write a file.
> • The operating system gives the permission to the program for operation on file.
> • Permission varies from read-only, read-write, denied and so on.
> • Operating System provides an interface to the user to create/delete files.
> • Operating System provides an interface to the user to create/delete directories.
> • Operating System provides an interface to create the backup of file system.

Question sources

Computer Science MCQ practice collection · Question 660

Question 134 | AM0661

What is the purpose of virtual memory in an operating system?

A. To provide a virtualized environment for running multiple operating systems

B. To increase the physical memory capacity of a computer system

C. To allow the execution of programs without any memory constraints

D. To create a separate memory space for each running process

Hint

Separate a permitted nonresident access from an illegal access and trace the required handling steps.

Answer D: Provide translated, protected per-process address spaces.

OS32 defines virtual memory as translated/protected address spaces with nonresident portions backed by storage. D is the option matching separate process memory spaces. A describes machine virtualization; B falsely adds physical RAM and C falsely removes all memory constraints. Isolation and translation are the supported purpose, with backing storage providing flexibility within finite resource limits.

Why each choice is right or wrong

A - Incorrect: Running multiple guest OSs describes machine virtualization, explicitly distinguished from process virtual memory.

B - Incorrect: Virtual memory does not physically add RAM; it maps address spaces using existing RAM and backing storage.

C - Incorrect: The note explicitly retains capacity constraints; virtual memory does not make execution memory-unlimited.

D - Correct: Separate translated/protected process address spaces match the virtual-memory purpose described here.

Primary note proof

Study-note section OS32

> Virtual memory provides translated, protected address spaces and allows nonresident portions of a program to be backed by storage. It does not add physical RAM or remove all capacity constraints.

Study-note section OS04

> Virtualization presents a virtual machine environment, often allowing guest operating systems on one host through a hypervisor. It differs from virtual memory, which translates a process's addresses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 37 | lines 6-11

> Virtual memory is a technique that allows the execution of processes which are not completely available in memory. The main visible advantage of this scheme is that programs can be larger than physical memory. Virtual memory is the separation of user logical memory from physical memory.
> This separation allows an extremely large virtual memory to be provided for programmers when only a smaller physical memory is available. Following are the situations, when entire

Question sources

Computer Science MCQ practice collection · Question 661

Question 135 | AM0662

What is the role of the shell in an operating system?

A. To manage and allocate system resources

B. To provide a graphical user interface for interacting with the system

C. To act as an interface between the user and the operating system kernel

D. To schedule and prioritize the execution of processes

Hint

Distinguish command interfaces, privileged core services and the stages of booting.

Answer C: To act as an interface between the user and the operating system kernel

The shell takes user commands and invokes kernel functions (PDF p54), making C the exact interface relationship. A and D describe other core functions; B imposes a GUI-only restriction absent from the command-interpreter definition.

Why each choice is right or wrong

A - Incorrect: System resources are the kernel/OS management role, not the shell's defining interface.

B - Incorrect: The module describes command interpretation, not a requirement that the shell be graphical.

C - Correct: Directly matches the user-to-kernel interface (p54).

D - Incorrect: CPU ordering is scheduling, not the shell definition.

Primary note proof

Study-note section OS03

> The kernel implements core resource management, protection and system-call handling. A shell reads/interprets user commands; a GUI is another interface. Browsers, office tools and editors are applications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 54 | lines 18-20

> commands of the operating system. It can be used to do various types of operations,
> call application programs etc.
> Security - Linux provides user security using authentication features like password

Operating System module | PDF page 54 | lines 24-30

> Hardware layer - Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc.).
> Kernel - Core component of Operating System, interacts directly with hardware,
> provides low level services to upper layer components.
> Shell - An interface to kernel, hiding complexity of kernel's functions from users. Takes

commands from user and executes kernel's functions.
Utilities - Utility programs giving user most of the functionalities of an operating systems.

Question sources

Computer Science MCQ practice collection · Question 662

Question 136 | AM0663

What is the primary purpose of process synchronization when processes perform conflicting accesses to shared state?

A. To prevent deadlock situations

B. To ensure fair allocation of system resources

C. To enable parallel execution of multiple processes

D. To coordinate shared-state access and required operation ordering safely

Hint

Identify conflicting shared accesses and the safety or progress property required.

Answer D: To coordinate shared-state access and required operation ordering safely

Conflicting accesses need coordination so interleaving does not corrupt shared state or violate required operation order. Critical-section protection provides mutual exclusion, with progress and bounded waiting required under the stated solution assumptions. Synchronization does not automatically eliminate deadlock; inconsistent lock acquisition can introduce it. It may also serialize work rather than create parallelism, so safe shared-access coordination is the precise primary purpose.

Why each choice is right or wrong

A - Incorrect: Locking can itself deadlock, so universal deadlock prevention is not the primary definition.

B - Incorrect: General resource-allocation fairness is broader than coordinating shared-state operations.

C - Incorrect: Synchronization can serialize conflicting operations and does not itself create parallel execution.

D - Correct: Coordinating shared accesses and required ordering safely captures the primary purpose.

Primary note proof

Study-note section OS10

A race occurs when results depend on uncoordinated interleaving of conflicting accesses. A critical section is code accessing shared state that needs protection, not the resource itself. A

correct exclusion solution requires mutual exclusion, progress and bounded waiting under stated assumptions. Mutual exclusion is a desired safety property; races, starvation and deadlock are possible failures.

Study-note section OS10

Keep critical regions small and ensure every conflicting access uses the same protection. A barrier waits until a defined set of participants reaches a point; a monitor encapsulates protected operations with condition synchronization. Locking can itself introduce deadlock if acquisition order is inconsistent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

Question sources

Computer Science MCQ practice collection · Question 663

Question 137 | AM0664

What is the main function of an operating system in terms of convenience?

A. Providing an easy-to-use graphical user interface

B. Optimizing program execution speed

C. Managing system resources efficiently

D. Ensuring data security and protection

Hint

Separate broad resource-management goals from individual services or applications.

Answer A: An easy-to-use interface is the convenience example.

OS01 defines convenience as making tasks easy through consistent interfaces and abstractions. A directly exemplifies that user-facing goal, although a GUI is one possible interface rather than a universal requirement. B and C concern speed/resource efficiency, and D protection. Those are worthwhile OS objectives but do not match the convenience dimension specifically named in the stem.

Why each choice is right or wrong

A - Correct: An easy-to-use GUI is an example of making user tasks convenient through an interface.

B - Incorrect: Optimizing execution speed concerns performance rather than the user-facing convenience dimension requested.

C - Incorrect: Efficient allocation is a resource-use objective, distinguished from convenience in the note.

D - Incorrect: Security/protection is a separate OS objective rather than the interface-usability example of convenience.

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process. Application development tools use OS services but application development itself is not a core resource-management function.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 5 | lines 6-13

> An operating System (OS) is an intermediary between users and computer hardware. It provides users an environment in which a user can execute programs conveniently and efficiently. In technical terms, it is software which manages hardware. An operating System controls the allocation of resources and services such as memory, processors, devices and information.
> 1.1 Definition
> An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Operating System module | PDF page 14 | lines 18-23

> Interactivity refers that a User is capable to interact with computer system. Operating system does the following activities related to interactivity.
> • OS provides user an interface to interact with system.
> • OS managers input devices to take inputs from the user. For example, keyboard.
> • OS manages output devices to show outputs to the user. For example, Monitor.
> • OS Response time needs to be short since the user submits and waits for the result.

Question sources

Computer Science MCQ practice collection · Question 664

Question 138 | AM0665

How does an operating system improve efficiency in a computer system?

A. By allocating system resources to processes effectively

B. By providing a wide range of software applications

C. By supporting multiple programming languages

D. By ensuring data integrity and security

Hint

Separate broad resource-management goals from individual services or applications.

Answer A: By allocating system resources to processes effectively

Efficient use follows the module's allocation of processor time and memory to processes (PDF pp5-6,31). A states this mechanism rather than the quantity of applications or languages.

Why each choice is right or wrong

A - Correct: Directly matches resource allocation and scheduling responsibilities.

B - Incorrect: More applications alone does not establish allocation efficiency.

C - Incorrect: Language variety alone does not establish allocation efficiency.

D - Incorrect: Security is a distinct access-control function, not the stated allocation mechanism.

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 5 | lines 6-13

> An operating System (OS) is an intermediary between users and computer hardware. It provides users an environment in which a user can execute programs conveniently and efficiently. In technical terms, it is software which manages hardware. An operating System controls the allocation of resources and services such as memory, processors, devices and information.
> 1.1 Definition
> An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Operating System module | PDF page 6 | lines 4-9

In multiprogramming environment, OS decides which process gets the processor when and how much time. This function is called process scheduling. Operating System does the following activities for processor management. Keeps tracks of processor and status of process. Program responsible for this task is known as traffic controller.
Allocates the processor (CPU) to a process.
De-allocates processor when processor is no longer required.

Operating System module | PDF page 31 | lines 24-31

Memory management is the functionality of an operating system which handles or manages primary memory. Memory management keeps track of each and every memory location either it is allocated to some process or it is free. It checks how much memory is to be allocated to processes. It decides which process will get memory at what time. It tracks whenever some memory gets freed or unallocated and correspondingly it updates the status.
Memory management provides protection by using two registers, a base register and a limit register. The base register holds the smallest legal physical memory address and the limit register specifies the size of the range. For example, if the base register holds 300000 and the

Question sources

Computer Science MCQ practice collection · Question 665

Question 139 | AM0666

What does it mean for an operating system to have the ability to evolve?

A. It can adapt to changing hardware configurations

B. It can support multiple operating systems simultaneously

C. It can run legacy software without compatibility issues

D. It can automatically update itself with new features and bug fixes

Hint

Separate broad resource-management goals from individual services or applications.

Answer A: Adapt to changing hardware configurations.

OS01 defines ability to evolve as adding hardware support, features and repairs through maintainable interfaces/modules. Adaptation to hardware configurations is therefore the direct example. Running multiple guest OSs describes virtualization. Backward compatibility preserves supported behavior rather than guaranteeing every legacy binary, and automatic updating is not a defining property of all operating systems.

Why each choice is right or wrong

A - Correct: Adding support for changing hardware is a direct example of maintainable OS evolution.

B - Incorrect: Supporting multiple guest operating systems describes virtualization rather than the definition of ability to evolve.

C - Incorrect: The absolute absence of compatibility issues is not guaranteed by evolution or supported backward compatibility.

D - Incorrect: Automatic updates may be provided, but the note explicitly rejects them as a defining property of every OS.

Primary note proof

Study-note section OS01

> Ability to evolve means adding hardware support, features and repairs through maintainable interfaces/modules. Backward compatibility preserves supported older application behavior; it is not a promise that every old binary works.

Study-note section OS04

> Virtualization presents a virtual machine environment, often allowing guest operating systems on one host through a hypervisor.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 8 | lines 28-31

> • Centralized servers are highly stable.
> • Security is server managed.
> • Upgrades to new technologies and hardware can be easily integrated into the system.
> • Remote access to servers is possible from different locations and types of systems.

Operating System module | PDF page 54 | lines 5-17

> Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
> Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
> Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
> Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
> Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
> Shell - Linux provides a special interpreter program which can be used to execute

Question sources

Computer Science MCQ practice collection · Question 666

Question 140 | AM0667

Which file-management feature directly helps users locate files and navigate among organized groups of files?

A. By providing advanced encryption techniques for file security

B. By allowing users to organize files in a hierarchical structure

C. By automatically backing up files to remote servers

D. By optimizing file storage to maximize disk space utilization

Hint

Follow namespace structure and distinguish names, underlying files and mounted filesystems.

Answer B: By allowing users to organize files in a hierarchical structure

A hierarchical directory tree gives users nested groups and paths for locating files and navigating between directories. This directly supplies the requested organizational convenience. Encryption protects confidentiality, backups preserve additional copies for recovery, and storage optimization concerns space utilization. Those functions may also be useful, but they do not themselves define the navigable directory organization asked for here.

Why each choice is right or wrong

A - Incorrect: Encryption protects confidentiality; it does not supply nested directory paths for navigation.

B - Correct: Hierarchical directories organize files into nested groups reachable through paths.

C - Incorrect: Remote backup concerns additional stored copies, rather than the structure used to navigate the working file namespace.

D - Incorrect: Maximizing disk-space utilization is an allocation goal rather than a directory-navigation feature.

Primary note proof

Study-note section OS37

> A single-level directory requires unique names in one namespace. Two-level directories separate users. A tree allows nested directories and absolute/relative paths; links can create shared graphs.

Study-note section OS39

> Encryption protects confidentiality under correct key use but does not automatically provide integrity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 6 | lines 19-23

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Operating System does the following activities for file management.
Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.

Operating System module | PDF page 54 | lines 5-17

Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
Shell - Linux provides a special interpreter program which can be used to execute

Question sources

Computer Science MCQ practice collection · Question 667

Question 141 | AM0668

What is the role of device drivers in an operating system's convenience factor?

A. They allow users to customize the appearance of the graphical user interface.

B. They provide compatibility with a wide range of hardware devices.

C. They optimize system performance for resource-intensive applications.

D. They enhance data security and encryption for sensitive files.

Hint

Separate device hardware, device-specific software and general event or transfer mechanisms.

Answer B: They provide compatibility with a wide range of hardware devices.

Drivers hide peculiarities of device hardware (PDF p10); p47 even explains a terminal driver supporting differing brands. Standard interfaces on p45 let applications use differing controllers, matching B.

Why each choice is right or wrong

A - Incorrect: GUI appearance is not the device-specific I/O abstraction.

B - Correct: Device compatibility through hardware-specific implementation and common interfaces matches pp10,45,47.

C - Incorrect: Application performance may benefit, but optimizing all resource-intensive applications is not the driver definition.

D - Incorrect: Encryption is not the driver's general stated I/O responsibility.

Primary note proof

Study-note section OS34

A controller is device hardware exposing status, control and data registers. A driver converts OS requests into device-specific operations. Device independence and uniform naming hide brand-specific details; error handling reports/retries supported failures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 30-37

I/O subsystem comprised of I/O devices and their corresponding driver software. Drivers hides the peculiarities of specific hardware devices from the user as the device driver knows the peculiarities of the specific device.
Operating System manages the communication between user and device drivers.
The major activities of an operating system with respect to I/O Operation.
• I/O operation means read or write operation with any file or any specific I/O device.
• Program may require any I/O device while running.
• Operating system provides the access to the required I/O device when required.

Operating System module | PDF page 47 | lines 19-26

12.5 Device driver
Device driver is a program or routine developed for an I/O device. A device driver implements I/O operations or behaviors on a specific class of devices. For example a system supports one or a number of multiple brands of terminals, all slightly different terminals may have a single terminal driver. In the layered structure of I/O system, device driver lies between interrupt handler and device independent I/O software. The job of a device driver is following.
To accept request from the device independent software above it.
To see to it that the request is executed.

Operating System module | PDF page 45 | lines 28-33

Application I/O Interface represents the structuring techniques and interfaces for the operating system to enable I/O devices to be treated in a standard, uniform way. The actual differences lies kernel level modules called device drivers which are custom tailored to corresponding devices but show one of the standard interfaces to applications. The purpose of the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, such as the I/O system calls. Following are the characteristics of I/O

Question sources

Computer Science MCQ practice collection · Question 668

Question 142 | AM0669

How does an operating system improve efficiency in terms of process management?

A. By providing a comprehensive programming environment

B. By executing processes in parallel on multiple CPUs

C. By automatically compressing files to reduce storage space

D. By enabling users to schedule program execution at specific times

Hint

Compare interaction, CPU sharing and deadline requirements under the stated system model.

Answer B: By executing processes in parallel on multiple CPUs

PDF p29 says the kernel can simultaneously schedule threads from one process across multiple processors, while p27 explains parallel performance benefits. B is the relevant offered mechanism when suitable parallel work and CPUs exist.

Why each choice is right or wrong

A - Incorrect: A programming environment does not itself describe parallel process execution.

B - Correct: Scheduling work across multiple CPUs exploits parallelism (pp27,29).

C - Incorrect: File compression is unrelated to CPU process management here.

D - Incorrect: User-chosen execution times are not the multiprocessor-efficiency mechanism.

Primary note proof

Study-note section OS02

> A single CPU interleaves execution; multiple CPUs can execute work truly in parallel. Shared-memory multiprocessors do not require each CPU to have private main memory.

Study-note section OS08

> In many-to-one mapping, a blocking kernel call can block all user threads and one kernel execution context cannot use multiple CPUs simultaneously. One-to-one permits another kernel thread to run when one blocks. Many-to-many multiplexes user threads over several kernel threads.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 29 | lines 4-16

> In this case, thread management done by the Kernel. There is no thread management code in the application area. Kernel threads are supported directly by the operating system. Any application can be programmed to be multithreaded. All of the threads within an application are supported within a single process.
> The Kernel maintains context information for the process as a whole and for individuals' threads within the process. Scheduling by the Kernel is done on a thread basis. The Kernel

performs thread creation, scheduling and management in Kernel space. Kernel threads are generally slower to create and manage than the user threads.
Advantages
• Kernel can simultaneously schedule multiple threads from the same process on multiple processes.
• If one thread in a process is blocked, the Kernel can schedule another thread of the same process.

Operating System module | PDF page 27 | lines 6-14

A thread is a flow of execution through the process code, with its own program counter, system registers and stack. A thread is also called a light weight process. Threads provide a way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.
Each thread belongs to exactly one process and no thread can exist outside a process. Each thread represents a separate flow of control. Threads have been successfully used in implementing network servers and web server. They also provide a suitable foundation for parallel execution of applications on shared memory multiprocessors. Following figure shows

Question sources

Computer Science MCQ practice collection · Question 669

Question 143 | AM0670

In the context of operating systems, what is the meaning of scalability?

A. The ability to run on different hardware platforms

B. The ability to support multiple programming languages

C. The ability to handle increasing workloads and resources

D. The ability to provide a user-friendly interface

Hint

Separate broad resource-management goals from individual services or applications.

Answer C: Handle increasing workloads and resources.

OS01 directly defines scalability as handling greater workloads/resources. Running on different hardware platforms is portability, and providing an easy interface is convenience. Programming-language support does not itself establish either increasing capacity or workload handling. The wording in C matches the supplied definition without assuming scalability guarantees limitless growth or unchanged performance.

Why each choice is right or wrong

A - Incorrect: Adapting to other hardware platforms is portability, explicitly distinguished from scalability.

B - Incorrect: Supporting programming languages does not define the ability to handle growing workload or resources.

C - Correct: Handling increasing workloads and resources directly matches the note’s definition of scalability.

D - Incorrect: A user-friendly interface concerns convenience rather than scaling capacity as demand grows.

Primary note proof

Study-note section OS01

> Scalability means handling greater workload/resources; portability means adapting to other platforms.

Study-note section OS01

> Convenience means making tasks easy through consistent interfaces, files and device abstractions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 54 | lines 5-17

> Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
> Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
> Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
> Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
> Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
> Shell - Linux provides a special interpreter program which can be used to execute

Question sources

Computer Science MCQ practice collection · Question 670

Question 144 | AM0671

How does an operating system ensure convenience in terms of software installation and management?

A. By providing an intuitive software development environment

B. By offering automatic software updates and patches

C. By optimizing the execution speed of software applications

D. By allowing users to customize the appearance of the graphical user interface

Hint

Separate broad resource-management goals from individual services or applications.

Answer B: Automatic updates and patches can make software maintenance convenient.

OS01 defines convenience as making tasks easy and notes that updates help administration. Among the alternatives, offering automatic updates and patches directly reduces routine software-management effort. This is an example of a convenience feature, not a universal guarantee that every OS updates automatically. Development environments, speed optimization and appearance customization address different activities from installation/maintenance in the stem.

Why each choice is right or wrong

A - Incorrect: A development environment assists creating software rather than directly providing installation and maintenance convenience.

B - Correct: Automating updates and patches can reduce routine software-management work, though the feature is not universal.

C - Incorrect: Execution-speed optimization concerns performance, not the installation/maintenance convenience specifically requested.

D - Incorrect: Customizing GUI appearance changes presentation rather than automating software installation or maintenance tasks.

Primary note proof

Study-note section OS01

> Convenience means making tasks easy through consistent interfaces, files and device abstractions.

Study-note section OS01

> Updates and monitoring help administration, but automatic updates are not a defining property of every OS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 8 | lines 28-31

• Centralized servers are highly stable.
• Security is server managed.
• Upgrades to new technologies and hardware can be easily integrated into the system.
• Remote access to servers is possible from different locations and types of systems.

Operating System module | PDF page 54 | lines 5-17

Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
Shell - Linux provides a special interpreter program which can be used to execute

Question sources

Computer Science MCQ practice collection · Question 671

Question 145 | AM0672

What does it mean for an operating system to provide fault tolerance?

A. It continues correct service despite the specified failures it is designed to tolerate

B. It can handle multiple user requests simultaneously.

C. It can optimize system performance for resource-intensive tasks.

D. It can provide a secure environment for sensitive data.

Hint

Separate broad resource-management goals from individual services or applications.

Answer A: It continues correct service despite the specified failures it is designed to tolerate

Fault tolerance means that correct service continues despite specified failures within the system's fault model. Recovery restores service after a failure and may be useful without providing uninterrupted correct service. Multiple-user handling concerns concurrency, resource-intensive performance concerns efficiency, and sensitive-data protection concerns security. The revised continuation option therefore states the specific property rather than substituting quick recovery for fault tolerance.

Why each choice is right or wrong

A - Correct: Continued correct service despite specified faults is the fault-tolerance property.

B - Incorrect: Simultaneous user requests concern concurrency rather than surviving failures.

C - Incorrect: Performance optimization does not define continued service under faults.

D - Incorrect: Security protects data but does not itself define fault tolerance.

Primary note proof

Study-note section OS01

> Fault tolerance continues correct service despite specified failures; recovery restores service afterward.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 8 | lines 15-16

> • If one site fails in a distributed system, the remaining sites can potentially continue operating.

Operating System module | PDF page 47 | lines 17-18

> Error Handling - An operating system that uses protected memory can guard against many kinds of hardware and application errors.

Question sources

Computer Science MCQ practice collection · Question 672

Question 146 | AM0673

How does an operating system support the ability to evolve in terms of software compatibility?

A. By providing a wide range of pre-installed software applications

B. By offering backward compatibility for legacy software

C. By optimizing the execution speed of software applications

D. By allowing users to customize system settings and preferences

Hint

Separate broad resource-management goals from individual services or applications.

Answer B: Backward compatibility for supported legacy software.

OS01 defines backward compatibility as preserving supported older application behavior. That is the software-compatibility aspect of evolution asked here. Preinstalled applications, execution speed and preference customization do not establish that older software's supported behavior remains available. The answer must retain the support qualification; it does not promise all legacy binaries run without any compatibility issues.

Why each choice is right or wrong

A - Incorrect: Preinstalling many applications does not establish compatibility with older applications or their supported behavior.

B - Correct: Backward compatibility preserves supported legacy behavior, directly matching the software-compatibility aspect of evolution.

C - Incorrect: Faster execution does not by itself preserve interfaces or behavior required by legacy software.

D - Incorrect: Customizing preferences changes configuration rather than establishing compatibility with older application behavior.

Primary note proof

Study-note section OS01

> Backward compatibility preserves supported older application behavior; it is not a promise that every old binary works.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 53 | lines 5-7

> Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

Operating System module | PDF page 54 | lines 5-17

> Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
> Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
> Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
> Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
> Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
> Shell - Linux provides a special interpreter program which can be used to execute

Question sources

Computer Science MCQ practice collection · Question 673

Question 147 | AM0674

What is deadlock in the standard process/resource model?

A. A state in which each member of a blocked set waits for an event or resource that only another blocked member can provide

B. It is a state where a process is terminated abruptly.

C. It is a state where multiple processes are executing concurrently.

D. It is a state where the operating system crashes due to a software bug.

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer A: A state in which each member of a blocked set waits for an event or resource that only another blocked member can provide

Deadlock describes a blocked set whose members depend on one another for the events or resources required to continue. Because each provider is itself blocked within the same dependency, normal execution cannot resolve the waits. A process being terminated, multiple processes executing, or an OS crash describes a different condition. The revised definition also distinguishes deadlock from starvation caused by unfair repeated bypass.

Why each choice is right or wrong

A - Correct: This states the closed dependency that defines the deadlocked set.

B - Incorrect: Abrupt termination ends execution rather than defining blocked dependencies.

C - Incorrect: Concurrent execution is normal and does not imply deadlock.

D - Incorrect: An OS crash is a different failure; deadlocked processes can exist while the OS remains active.

Primary note proof

Study-note section OS22

> A set is deadlocked when each member waits for an event/resource that only another blocked member can provide. Mere waiting for another holder is normal blocking; indefinite unfair delay can be starvation without a cycle of dependencies.

Study-note section OS06

> New processes are created/admitted; ready means eligible for CPU but not currently running. Running uses a CPU. Waiting/blocked means unable to proceed until an event; terminated means execution has ended.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

• Loads a program into memory.
• Executes the program.
• Handles program's execution.
• Provides a mechanism for process synchronization.
• Provides a mechanism for process communication.
• Provides a mechanism for deadlock handling.

Operating System module | PDF page 16 | lines 29-39

1. New The process is being created.
2.
Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
3.
Running Process instructions are being executed (i.e. The process that is currently being executed).
4.
Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
5. Terminated The process has finished execution.

Question sources

Computer Science MCQ practice collection · Question 674

Question 148 | AM0675

Which of the following conditions must be present for deadlock to occur?

A. Mutual exclusion, hold and wait, no preemption, and circular wait.

B. Mutual exclusion, hold and wait, preemption, and circular wait.

C. Mutual inclusion, hold and wait, no preemption, and circular wait.

D. Mutual inclusion, hold and wait, preemption, and circular wait.

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer A: Mutual exclusion, hold-and-wait, no-preemption and circular wait.

OS22 lists the four necessary conditions in the standard reusable-resource model. A preserves the exact set: exclusion, holding while requesting more, no forced resource withdrawal, and a wait cycle. B introduces preemption in place of no-preemption; C and D replace exclusion with inclusion. These are not harmless wording changes because they remove or reverse the relevant restrictions.

Why each choice is right or wrong

A - Correct: This option states all four standard necessary conditions with their correct meanings and polarity.

B - Incorrect: It substitutes preemption for no-preemption, reversing the required resource-withdrawal restriction.

C - Incorrect: It substitutes mutual inclusion for mutual exclusion, losing the nonshareable-resource condition.

D - Incorrect: It reverses both exclusion and no-preemption, so it is not the required condition set.

Primary note proof

Study-note section OS22

> Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Study-note section OS22

> All four are necessary in the standard reusable-resource model. Their availability as general system policies does not prove a deadlock exists at this instant; inspect actual allocations/requests.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Computer Science MCQ practice collection · Question 675

Question 149 | AM0676

What is mutual exclusion in the context of deadlock?

A. It is the condition where multiple processes can access the same resource simultaneously.

B. It is the condition where a resource can be accessed by only one process at a time.

C. It is the condition where a process holds a resource while waiting for another resource.

D. It is the condition where a process waits indefinitely for a resource.

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer B: At most one process holds the nonshareable resource at a time.

OS22 defines mutual exclusion as allowing at most one holder of a nonshareable resource. B states that one-at-a-time condition. A describes concurrent sharing, C describes holding while requesting another resource, and D describes indefinite waiting without establishing exclusivity. The definition concerns access safety, not a guarantee that every waiting process eventually obtains the resource.

Why each choice is right or wrong

A - Incorrect: Simultaneous access by several processes contradicts the exclusive-holder condition for a nonshareable resource.

B - Correct: Only one process at a time may hold or use the nonshareable resource.

C - Incorrect: Retaining one resource while requesting another defines hold-and-wait rather than mutual exclusion.

D - Incorrect: Indefinite waiting describes a possible failure outcome and does not define exclusive resource possession.

Primary note proof

Study-note section OS22

> Mere waiting for another holder is normal blocking; indefinite unfair delay can be starvation without a cycle of dependencies. Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Computer Science MCQ practice collection · Question 676

Question 150 | AM0677

What is hold and wait in the context of deadlock?

A. It is the condition where a process holds a resource while waiting for another resource.

B. It is the condition where a process waits indefinitely for a resource.

C. It is the condition where multiple processes can access the same resource simultaneously.

D. It is the condition where a resource can be accessed by only one process at a time.

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer A: Retain a resource while waiting for another.

OS22 directly defines hold-and-wait as retaining a resource while requesting another. A contains both parts of that definition. Merely waiting indefinitely does not establish a retained allocation, simultaneous sharing contradicts exclusion, and one-at-a-time access describes mutual exclusion rather than a process holding one resource while requesting more.

Why each choice is right or wrong

A - Correct: The process simultaneously retains an allocated resource and waits to acquire a further one.

B - Incorrect: Indefinite waiting does not establish that the process still holds another resource.

C - Incorrect: Simultaneous shared access is not the retained-allocation-plus-request condition defining hold-and-wait.

D - Incorrect: One process accessing a resource at a time defines mutual exclusion, not hold-and-wait.

Primary note proof

Study-note section OS22

> Mere waiting for another holder is normal blocking; indefinite unfair delay can be starvation without a cycle of dependencies. Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.

- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

Question sources

Computer Science MCQ practice collection · Question 677

Question 151 | AM0678

What is preemption in the context of deadlock prevention?

A. It is the condition where a process holds a resource while waiting for another resource.

B. It is the condition where a process waits indefinitely for a resource.

C. It is the condition where multiple processes can access the same resource simultaneously.

D. It is the condition where a resource can be forcibly taken from a process.

Hint

Identify the resource-request restriction and which necessary condition it removes.

Answer D: A resource may be forcibly taken from its holder.

OS22 defines no-preemption as voluntary release only, while OS24 permits safe withdrawal/restart as a way to attack that restriction. Preemption is the opposite permission: a resource can be forcibly reclaimed. D directly states it. The answer concerns resources that can safely be withdrawn, not a claim that every device or lock can be preempted without corruption.

Why each choice is right or wrong

A - Incorrect: Holding one resource while waiting for another describes hold-and-wait rather than forced resource withdrawal.

B - Incorrect: Indefinite waiting is a possible outcome, not the permission to reclaim a held resource.

C - Incorrect: Concurrent sharing concerns resource shareability rather than withdrawing an existing allocation from its holder.

D - Correct: Forcibly taking a held resource is resource preemption, subject to safe withdrawal/restart support.

Primary note proof

Study-note section OS24

Prevention constrains how requests are made. Share a resource where safe to remove mutual exclusion; request all needed resources before holding any to remove hold-and-wait; permit safe resource withdrawal/restart to attack no-preemption; impose a global acquisition order to remove circular wait. Not every physical resource is shareable or safely preemptable.

Study-note section OS22

Mere waiting for another holder is normal blocking; indefinite unfair delay can be starvation without a cycle of dependencies. Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

• Loads a program into memory.
• Executes the program.
• Handles program's execution.
• Provides a mechanism for process synchronization.
• Provides a mechanism for process communication.
• Provides a mechanism for deadlock handling.

Operating System module | PDF page 19 | lines 6-12

A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
• The process could issue an I/O request and then it would be placed in an I/O queue.
• The process could create new sub process and will wait for its termination.
• The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Question sources

Computer Science MCQ practice collection · Question 678

Question 152 | AM0679

What is circular wait in the context of deadlock?

A. It is the condition where a process waits indefinitely for a resource.

B. It is the condition where multiple processes can access the same resource simultaneously.

C. It is the condition where a process holds a resource while waiting for another resource.

D. It is the condition where a circular chain of processes exists, each holding a resource requested
by the next process in the chain.

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer D: A closed chain of resource waits.

OS22 defines circular wait as a closed chain of waits. D specifies that each process holds a resource requested by the next in a circular chain, which returns the dependency to the starting participant. C describes only one hold-and-wait relationship, which need not be cyclic. Indefinite waiting and simultaneous sharing likewise fail to establish a closed chain.

Why each choice is right or wrong

A - Incorrect: Indefinite waiting does not identify a closed cycle of dependencies among resource holders.

B - Incorrect: Simultaneous resource sharing is not a chain of blocked requests and does not define circular wait.

C - Incorrect: One process holding while requesting another establishes hold-and-wait but not necessarily a cycle.

D - Correct: The circular chain of holdings and requests explicitly forms the required closed wait dependency.

Primary note proof

Study-note section OS22

> Mere waiting for another holder is normal blocking; indefinite unfair delay can be starvation without a cycle of dependencies. Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Computer Science MCQ practice collection · Question 679

Question 153 | AM0680

Which listed policy does not actively prevent an unsafe allocation or detect and recover from an existing deadlock? Interpret “deadlock acceptance” as deliberately ignoring sufficiently rare deadlocks.

A. Deadlock avoidance

B. Deadlock detection and recovery

C. Deadlock prevention

D. Deadlock acceptance

Hint

Separate handling before allocation from actions taken after detection, and consider the cost of checking.

Answer D: Deadlock acceptance

Prevention constrains requests to break a necessary condition; avoidance keeps tentative allocations safe; detection and recovery identifies and resolves an existing deadlock. Deliberate acceptance, as defined here, instead ignores a sufficiently rare problem rather than actively applying those controls. It is still a possible policy choice, but is the only listed one outside the active-control category specified by the revised stem.

Why each choice is right or wrong

A - Incorrect: Avoidance actively restricts grants to preserve a safe state.

B - Incorrect: Detection and recovery actively identifies and resolves deadlock.

C - Incorrect: Prevention actively removes a necessary deadlock condition.

D - Correct: Intentional ignoring is a policy choice but does not actively apply the specified controls.

Primary note proof

Study-note section OS26

> Detection asks whether the current state is deadlocked. Recovery can terminate selected processes or reclaim safely preemptable resources and roll affected work back to a checkpoint. Termination and rollback are both possible recovery tools; choose according to the resource and process conditions.

Study-note section OS26

> Ignoring a sufficiently rare problem is sometimes called the ostrich policy, but real systems still prevent or detect selected deadlocks.

Study-note section OS24

> Prevention constrains how requests are made.

Study-note section OS25

Avoidance allows requests only if the resulting state remains safe.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

Question sources

Computer Science MCQ practice collection · Question 680

Question 154 | AM0681

In a resource-allocation graph with one instance of each resource type, which listed roles are valid? Read B as direct resource-to-resource edges and D as showing permission to forcibly withdraw resources. Select all that apply.

A. To represent the allocation of resources to processes.

B. To represent the dependencies between resources.

C. To represent the circular wait condition.

D. To represent the preemption of resources.

Hint

Follow request and assignment arrows and check how many instances each resource has.

Answer A, C: To represent the allocation of resources to processes.; To represent the circular wait condition.

An assignment edge records which process holds a resource instance, and a request edge records what a process is waiting for. Following alternating edges can form a circular wait. In the stated single-instance model a cycle establishes deadlock. The graph does not use direct resource-to-resource dependency edges and does not encode a general permission to preempt resources, so A and C are the valid listed roles.

Why each choice is right or wrong

A - Correct: Assignment edges represent resource holdings by processes.

B - Incorrect: Standard graph edges connect processes and resources, not resources directly to other resources.

C - Correct: A cycle of request and assignment edges represents circular wait.

D - Incorrect: Resource-preemption permission is not encoded by the stated request/assignment edges.

Primary note proof

Study-note section OS23

> In a resource-allocation graph, P→R is a request and an instance R→P is an assignment. Follow these directions to form waits. No cycle implies no deadlock; with one instance of every resource type, a cycle implies deadlock. With multiple instances a cycle alone is insufficient: another instance may permit completion. A wait-for graph replaces single-instance resource nodes with P→Q when P waits for Q.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Question sources

Computer Science MCQ practice collection · Question 681

Question 155 | AM0682

Which resource allocation policy can help prevent deadlock?

A. First-Come, First-Served (FCFS)

B. Round Robin (RR)

C. Shortest Job Next (SJN)

D. Banker's Algorithm

Hint

Compute remaining needs and simulate feasible completions before deciding whether an allocation is safe.

Answer D: Banker’s algorithm, technically an avoidance policy.

OS25 states that Banker grants requests only while preserving a safe state. That can prevent entry into a deadlocked state in the broad everyday sense used by can help prevent, although its technical category is avoidance rather than condition-breaking prevention. FCFS, RR and SJN are CPU scheduling policies and do not themselves test resource-allocation safety.

Why each choice is right or wrong

A - Incorrect: FCFS orders CPU bursts by arrival rather than testing maximum resource claims and safe allocations.

B - Incorrect: RR schedules CPU turns; time slicing does not establish safe allocation of locks or other resources.

C - Incorrect: SJN selects shortest next CPU bursts rather than performing a resource-state safety test.

D - Correct: Banker admits resource requests only when the resulting state is safe; its strict classification is avoidance.

Primary note proof

Study-note section OS25

> Avoidance allows requests only if the resulting state remains safe. Banker requires maximum claims. Compute Need=Max−Allocation.

Study-note section OS14

> Non-preemptive scheduling keeps a running CPU burst until completion, blocking or voluntary yielding. Preemptive scheduling can return running work to ready, usually after a timer quantum or a more urgent arrival. FCFS and non-preemptive SJF are non-preemptive; RR and SRTF are preemptive. Priority can be either, so its name alone cannot decide. Cooperative describes relying on voluntary release.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

> • Loads a program into memory.
> • Executes the program.
> • Handles program's execution.
> • Provides a mechanism for process synchronization.
> • Provides a mechanism for process communication.
> • Provides a mechanism for deadlock handling.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.
> • Easy to understand and implement.
> • Poor in performance as average wait time is high.
> Wait time of each process is following
> Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.

• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Question sources

Computer Science MCQ practice collection · Question 682

Question 156 | AM0683

What is the purpose of the Banker's Algorithm in deadlock avoidance?

A. To break the hold and wait condition.

B. To allocate resources in a fair and equitable manner.

C. To detect deadlocked processes and recover from deadlock.

D. To ensure that resource requests will not lead to an unsafe state.

Hint

Compute remaining needs and simulate feasible completions before deciding whether an allocation is safe.

Answer D: Ensure each granted request preserves a safe state.

Banker’s algorithm checks whether a tentative resource grant preserves a safe state. It computes remaining needs and simulates an order in which all processes can finish, granting only requests within the declared need and available resources that pass this safety test. It is an avoidance policy, not a fairness guarantee, an already-deadlocked-state detector, or prevention by eliminating hold-and-wait.

Why each choice is right or wrong

A - Incorrect: Banker does not require removing hold-and-wait; it checks whether tentative allocation remains safe.

B - Incorrect: Its safety criterion is not a guarantee of fair or equal allocation among processes.

C - Incorrect: Detection and recovery address actual deadlock, whereas Banker checks safety before granting a request.

D - Correct: Preserving a safe state after each grant is the purpose of the Banker avoidance test.

Primary note proof

Study-note section OS25

Avoidance allows requests only if the resulting state remains safe. Banker requires maximum claims. Compute Need=Max−Allocation. Set Work=Available and repeatedly find an unfinished process with Need≤Work componentwise; simulate completion by adding its Allocation to Work. Finishing all gives a safe sequence. Tentatively grant a request only if it is within Need and Available and the safety test succeeds. Unsafe does not mean already deadlocked. Banker is avoidance, not CPU scheduling, detection or prevention by breaking a condition.

Study-note section OS24

Prevention constrains how requests are made.

Study-note section OS26

Detection asks whether the current state is deadlocked. Recovery can terminate selected processes or reclaim safely preemptable resources and roll affected work back to a checkpoint.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

Question sources

Computer Science MCQ practice collection · Question 683

Question 157 | AM0684

Which of the following is a characteristic of preemptive scheduling?

A. Tasks cannot be interrupted once started.

B. Tasks are allowed to complete their execution.

C. Tasks can be interrupted and rescheduled.

D. Tasks are scheduled based on their arrival time.

Hint

Check whether the running burst can be interrupted before completion or blocking.

Answer C: Tasks can be interrupted and rescheduled.

PDF p25 explicitly preempts a process after its quantum; p19 returns forcibly removed work to the ready queue. C states interrupting and rescheduling that work.

Why each choice is right or wrong

A - Incorrect: Contradicts the explicit forced removal/preemption on pp19,25.

B - Incorrect: Tasks can eventually complete, but permission to finish does not distinguish preemptive scheduling.

C - Correct: Matches the interruption/rescheduling mechanism.

D - Incorrect: Arrival ordering defines FCFS (p23), not preemption.

Primary note proof

Study-note section OS14

> Non-preemptive scheduling keeps a running CPU burst until completion, blocking or voluntary yielding. Preemptive scheduling can return running work to ready, usually after a timer quantum or a more urgent arrival. FCFS and non-preemptive SJF are non-preemptive; RR and SRTF are preemptive. Priority can be either, so its name alone cannot decide. Cooperative describes relying on voluntary release.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 19 | lines 6-12

> A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
> • The process could issue an I/O request and then it would be placed in an I/O queue.
> • The process could create new sub process and will wait for its termination.
> • The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Question sources

Computer Science MCQ practice collection · Question 684

Question 158 | AM0685

Which listed scheduling policies are non-preemptive? SJN means non-preemptive shortest-next-burst scheduling; D specifically uses immediate preemption for a higher-priority arrival. Select all that apply.

A. Round Robin

B. First-Come, First-Served (FCFS)

C. Shortest Job Next (SJN)

D. Preemptive Priority Scheduling

Hint

Check whether the running burst can be interrupted before completion or blocking.

Answer B, C: First-Come, First-Served (FCFS); Shortest Job Next (SJN)

FCFS ordinarily serves each ready burst in arrival order without forcibly taking the CPU away. SJN, as explicitly defined, chooses the shortest ready burst and then lets it continue non-preemptively. RR can interrupt a burst at its quantum boundary, while the stated priority variant interrupts for a higher-priority arrival. Thus FCFS and SJN are the two non-preemptive policies in this list.

Why each choice is right or wrong

A - Incorrect: RR uses quantum expiry to preempt unfinished work.

B - Correct: FCFS ordinarily lets the selected burst continue without forced preemption.

C - Correct: SJN is explicitly the non-preemptive shortest-next-burst policy.

D - Incorrect: This option explicitly preempts for a higher-priority arrival.

Primary note proof

Study-note section OS14

> Non-preemptive scheduling keeps a running CPU burst until completion, blocking or voluntary yielding. Preemptive scheduling can return running work to ready, usually after a timer quantum or a more urgent arrival. FCFS and non-preemptive SJF are non-preemptive; RR and SRTF are preemptive. Priority can be either, so its name alone cannot decide. Cooperative describes relying on voluntary release.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job. Check whether a smaller or larger numeric value means higher priority. State whether a high-priority arrival preempts.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Question sources

Computer Science MCQ practice collection · Question 685

Question 159 | AM0686

Which scheduling algorithm is based on the priority assigned to each task?

A. Round Robin

B. First-Come, First-Served (FCFS)

C. Shortest Job Next (SJN)

D. Priority Scheduling

Hint

Check priority direction, preemption and whether waiting can change effective rank.

Answer D: Priority Scheduling

PDF p24 assigns a priority to each process and executes the highest priority first. That is D; other policies use quantum, arrival or execution length.

Why each choice is right or wrong

A - Incorrect: RR uses fixed quantum (p25).

B - Incorrect: FCFS follows arrival order (p23).

C - Incorrect: SJN uses job length (p23).

D - Correct: Assigned priority is the explicit selection rule (p24).

Primary note proof

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 24 | lines 10-15

> 7.3 Priority Based Scheduling
> • Each process is assigned a priority. Process with highest priority is to be executed first and so on.
> • Processes with same priority are executed on first come first serve basis.
> • Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.

• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Question sources

Computer Science MCQ practice collection · Question 686

Computer Science MCQ practice collection · Question 697

Question 160 | AM0687

User A has four continuously ready tasks and user B has one. The policy goal is equal aggregate CPU entitlement for each user, independent of task count. Which listed policy directly implements that goal when configured with equal per-user weights?

A. Round Robin

B. First-Come, First-Served (FCFS)

C. Shortest Job Next (SJN)

D. Fair-Share Scheduling

Hint

Identify the allocation entity, entitlement weights and whether the guarantee is probabilistic.

Answer D: Fair-Share Scheduling

Equal-turn RR gives each of the five tasks roughly one fifth of CPU service, so user A receives about 80 percent and B about 20 percent. Equal per-user fair-share instead gives each user half, dividing A’s half among its four tasks. FCFS and SJN do not enforce that user entitlement. The explicitly defined participant scope makes fair-share the unique matching policy.

Why each choice is right or wrong

A - Incorrect: Per-task RR gives A four fifths of service rather than equal per-user shares.

B - Incorrect: FCFS orders bursts by arrival and does not enforce per-user entitlements.

C - Incorrect: SJN favors short bursts and does not allocate equal user shares.

D - Correct: Equal per-user fair-share weights directly implement the stated half-and-half goal.

Primary note proof

Study-note section OS20

Fair-share scheduling allocates according to stated weights, potentially per user/group rather than per process. Equal per-process RR can let a user obtain more aggregate CPU by launching more processes. Proportional shares address that policy choice; fair need not mean equal.

Study-note section OS20

If user A runs four ready tasks and B runs one, plain RR gives A about 80% overall. A policy granting each user half the CPU instead gives A's tasks about 12.5% each and B's task 50%.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Question sources

Computer Science MCQ practice collection · Question 687

Question 161 | AM0688

A ready target has finitely many jobs ahead, all CPU bursts and RR quanta are finite and positive, context overhead is finite, and later arrivals join the tail without bypassing existing peers. Infinitely many later short or high-priority jobs may arrive over time. Which policies prevent that target from being postponed forever by later arrivals? D uses fixed priorities without aging. Select all that apply.

A. Round Robin

B. First-Come, First-Served (FCFS)

C. Shortest Job Next (SJN)

D. Fixed-priority scheduling without aging

Hint

Follow arrival order, include idle periods and distinguish waiting from completion time.

Answer A, B: Round Robin; First-Come, First-Served (FCFS)

RR gives the target a turn after the finite work ahead has received bounded turns. FCFS serves the finite earlier bursts before later arrivals, so the target eventually starts. SJN can repeatedly prefer later short jobs, and fixed priority can repeatedly prefer later high-priority jobs. The assumptions distinguish guaranteed eventual service from policies that permit an endless stream to bypass the target.

Why each choice is right or wrong

A - Correct: RR's finite cyclic turns reach the target without later-arrival bypass.

B - Correct: FCFS drains the finite predecessor work before serving later arrivals.

C - Incorrect: An endless sequence of short arrivals can keep a long target waiting under SJN.

D - Incorrect: An endless sequence of higher-priority arrivals can starve a low-priority target without aging.

Primary note proof

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt. If the CPU is idle, jump to the next arrival. Long CPU-bound work can make many short jobs wait: the convoy effect. A finite burst with finitely many jobs ahead eventually starts, but waiting can still be large.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Study-note section OS17

> Fixed-priority scheduling can starve low-priority jobs under continued high-priority load. SJF can also starve long jobs; starvation is not exclusive to fixed-priority scheduling.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Question sources

Computer Science MCQ practice collection · Question 688

Question 162 | AM0689

Which scheduling algorithm is based on the estimated remaining time required for each task?

A. Round Robin

B. First-Come, First-Served (FCFS)

C. Shortest Job Next (SJN)

D. Shortest Remaining Time First (SRTF)

Hint

Compare the ready jobs using the stated burst or remaining-time rule.

Answer D: Shortest Remaining Time First (SRTF).

OS16 says SRTF reconsiders on arrivals and selects the shortest remaining work. This matches the estimated remaining-time criterion in the stem. SJN chooses the shortest next burst non-preemptively, while RR uses cyclic quanta and FCFS arrival order. The distinction matters after execution has begun: the originally shortest burst need not be the currently shortest remaining job.

Why each choice is right or wrong

A - Incorrect: RR rotates bounded turns rather than selecting the job with the smallest estimated remaining execution.

B - Incorrect: FCFS follows arrival order rather than comparing estimated remaining CPU work.

C - Incorrect: Non-preemptive SJN compares next burst lengths when selecting, not continuously reconsidered remaining work.

D - Correct: SRTF explicitly selects the shortest remaining work and can preempt on a shorter arrival.

Primary note proof

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs. SRTF reconsiders on arrivals and selects the shortest remaining work; a shorter newcomer can preempt. Future burst lengths normally need estimates.

Study-note section OS16

> Repeated short arrivals can starve a long job. The shortest total original burst is not necessarily the shortest remaining burst after execution has begun.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 23-26

> 7.2 Shortest Job First (SJF)
> • Best approach to minimize waiting time.
> • Impossible to implement
> • Processer should know in advance how much time process will take.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science MCQ practice collection · Question 689

Question 163 | AM0690

Which policies can starve a ready job through repeated preference for later arrivals? Assume correct implementations, finite bursts, ordinary RR with a positive quantum, FCFS without bypass, non-preemptive SJN, and fixed-priority scheduling without aging. Select all that apply.

A. Round Robin

B. First-Come, First-Served (FCFS)

C. Shortest Job Next (SJN)

D. Priority Scheduling

Hint

Check priority direction, preemption and whether waiting can change effective rank.

Answer C, D: Shortest Job Next (SJN); Priority Scheduling

SJN can continually choose newly arrived short bursts while an older long job remains ready. Fixed priority without aging can likewise keep choosing high-priority arrivals over a low-priority target. Ordinary RR cycles peers, and FCFS does not let later arrivals overtake a finite set of predecessors. Thus the preference rules of SJN and fixed priority create the two stated starvation risks.

Why each choice is right or wrong

A - Incorrect: Ordinary RR returns ready jobs to recurring turns rather than repeatedly bypassing a peer.

B - Incorrect: FCFS preserves arrival order, so later arrivals do not overtake the target.

C - Correct: Repeated short arrivals can starve an older long SJN job.

D - Correct: Continued high-priority arrivals can starve a low-priority job without aging.

Primary note proof

Study-note section OS17

Fixed-priority scheduling can starve low-priority jobs under continued high-priority load. SJF can also starve long jobs; starvation is not exclusive to fixed-priority scheduling.

Study-note section OS15

FCFS runs ready bursts in arrival order and ordinarily does not preempt. If the CPU is idle, jump to the next arrival. Long CPU-bound work can make many short jobs wait: the convoy effect. A finite burst with finitely many jobs ahead eventually starts, but waiting can still be large.

Study-note section OS18

RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Question sources

Computer Science MCQ practice collection · Question 690

Question 164 | AM0691

Which scheduling algorithm uses time slices to allocate CPU time to tasks?

A. Round Robin

B. First-Come, First-Served (FCFS)

C. Shortest Job Next (SJN)

D. Priority Scheduling

Hint

Trace bounded turns in ready-queue order and include the stated overhead and arrivals.

Answer A: Round Robin

Each RR process receives a fixed time quantum (PDF p25); it is preempted when that period expires. A is the direct time-slice mechanism.

Why each choice is right or wrong

A - Correct: Exactly the fixed-quantum rule (p25).

B - Incorrect: Arrival-order service does not itself define repeated time slices (p23).

C - Incorrect: SJN bases selection on execution duration (p23).

D - Incorrect: Priority bases selection on assigned rank (p24); it need not use a quantum.

Primary note proof

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.
> • Easy to understand and implement.
> • Poor in performance as average wait time is high.
> Wait time of each process is following
> Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science MCQ practice collection · Question 691

Question 165 | AM0692

For a fixed finite set of simultaneous ready jobs with known CPU bursts, no I/O and zero switch cost, which listed non-preemptive policy minimizes average waiting over all possible execution orders for arbitrary burst lengths?

A. Round Robin

B. First-Come, First-Served (FCFS)

C. Shortest Job Next (SJN)

D. Shortest Remaining Time First (SRTF)

Hint

Compare the ready jobs using the stated burst or remaining-time rule.

Answer C: Shortest Job Next (SJN)

For the simultaneous known-burst workload, shortest-job order minimizes total waiting. Consider adjacent bursts L>S after an elapsed prefix T. Long then short gives waits T and T+L; short then long gives T and T+S, reducing their sum by L−S. Jobs outside the pair are unaffected. Repeatedly removing inversions gives SJN order. FCFS does not enforce it, while RR and SRTF are preemptive policies.

Why each choice is right or wrong

A - Incorrect: RR is preemptive and is outside the requested non-preemptive policy class.

B - Incorrect: FCFS can tie on favorable orders but does not minimize waiting for arbitrary burst orders.

C - Correct: SJN selects the shortest ready bursts and gives the standard minimum-average-waiting order.

D - Incorrect: SRTF is the preemptive remaining-time policy rather than the requested non-preemptive one.

Primary note proof

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs. SRTF reconsiders on arrivals and selects the shortest remaining work; a shorter newcomer can preempt. Future burst lengths normally need estimates. SJF minimizes average waiting for the usual fixed simultaneous ready set with known bursts and no preemption/switch cost; that theorem is not a workload-independent comparison against every preemptive policy.

Study-note section OS13

> For arrival A, first start S and completion C, response=S−A and turnaround=C−A. For a single CPU burst B with no I/O, waiting=C−A−B.

Study-note section OS14

> FCFS and non-preemptive SJF are non-preemptive; RR and SRTF are preemptive.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 23-26

> 7.2 Shortest Job First (SJF)
> • Best approach to minimize waiting time.
> • Impossible to implement
> • Processer should know in advance how much time process will take.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.
> • Easy to understand and implement.
> • Poor in performance as average wait time is high.
> Wait time of each process is following
> Process Wait Time: Service Time - Arrival Time

Question sources

Computer Science MCQ practice collection · Question 692

Question 166 | AM0693

Which scheduling algorithm provides better response time for interactive tasks?

A. Round Robin

B. First-Come, First-Served (FCFS)

C. Shortest Job Next (SJN)

D. Priority Scheduling

Hint

Trace bounded turns in ready-queue order and include the stated overhead and arrivals.

Answer A: Round Robin

Time-sharing seeks short response by giving users short CPU bursts (PDF p7). RR implements fixed quantum turns (p25), so A is a well-supported interactive choice with a suitably short quantum.

Why each choice is right or wrong

A - Correct: Repeated short quanta implement the p7 interactive strategy.

B - Incorrect: Arrival-order long work can delay later interactive work (p23).

C - Incorrect: SJN needs duration estimates (p23), not a guaranteed prompt periodic turn.

D - Incorrect: Priority may help selected interactive jobs, but depends on assigned ranks rather than the general time-sharing mechanism.

Primary note proof

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave. Specify how arrivals at a quantum boundary enter the queue. A large quantum approaches FCFS; tiny quanta increase switch overhead. RR gives recurring opportunities, not identical total CPU consumption for jobs of different lengths.

Study-note section OS21

> RR is a conventional interactive choice because turns improve responsiveness, subject to quantum and overhead.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 7 | lines 20-33

Time sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multi-programmed Batch Systems and Time-Sharing Systems is that in case of multi-programmed batch systems, objective is to maximize processor use, whereas in Time-Sharing Systems objective is to minimize response time.
Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, processor execute each user program in a short burst or quantum of computation. That is if n users are present, each user can get time quantum. When the user submits the command, the response time is in few seconds at most.
Operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science MCQ practice collection · Question 693

Computer Science MCQ practice collection · Question 700

Question 167 | AM0694

Which listed policy is preemptive? SJN denotes non-preemptive shortest-job-next, and D specifically keeps the running burst until completion, blocking or voluntary release.

A. First-Come, First-Served (FCFS)

B. Round Robin

C. Shortest Job Next (SJN)

D. Non-preemptive Priority Scheduling

Hint

Check whether the running burst can be interrupted before completion or blocking.

Answer B: Round Robin

RR limits a turn to a quantum and can return an unfinished runnable job to the ready queue. That is preemption. FCFS and the stated SJN variant keep a selected burst until it ends or blocks. The priority option is explicitly non-preemptive, so a higher-priority arrival does not interrupt its current burst. RR is therefore the unique preemptive policy in this list.

Why each choice is right or wrong

A - Incorrect: FCFS ordinarily does not forcibly interrupt the selected burst.

B - Correct: RR preempts unfinished work at quantum expiry.

C - Incorrect: SJN is explicitly the non-preemptive shortest-job policy.

D - Incorrect: The stated priority variant is explicitly non-preemptive.

Primary note proof

Study-note section OS14

> Non-preemptive scheduling keeps a running CPU burst until completion, blocking or voluntary yielding. Preemptive scheduling can return running work to ready, usually after a timer quantum or a more urgent arrival. FCFS and non-preemptive SJF are non-preemptive; RR and SRTF are preemptive. Priority can be either, so its name alone cannot decide. Cooperative describes relying on voluntary release.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science MCQ practice collection · Question 694

Question 168 | AM0695

Which listed policy allocates CPU according to declared user/group entitlement weights, so fairness can mean unequal proportional shares rather than arrival order?

A. First-Come, First-Served (FCFS)

B. Shortest Job Next (SJN)

C. Priority Scheduling

D. Fair-Share Scheduling

Hint

Identify the allocation entity, entitlement weights and whether the guarantee is probabilistic.

Answer D: Fair-Share Scheduling

Fair-share scheduling allocates service according to stated weights, which may apply to users or groups rather than individual processes. That directly implements proportional entitlement. FCFS respects arrival order, SJN chooses shorter bursts, and priority chooses a ranking; those rules do not themselves specify weighted user/group shares. The revised fairness criterion therefore identifies fair-share without implying every fair policy must use equal shares.

Why each choice is right or wrong

A - Incorrect: FCFS provides arrival ordering rather than declared weighted entitlements.

B - Incorrect: SJN selects by burst length rather than entitlement weights.

C - Incorrect: Priority ranking does not itself implement proportional user/group shares.

D - Correct: Fair-share scheduling allocates according to declared weights and participant scope.

Primary note proof

Study-note section OS20

> Fair-share scheduling allocates according to stated weights, potentially per user/group rather than per process. Equal per-process RR can let a user obtain more aggregate CPU by launching more processes. Proportional shares address that policy choice; fair need not mean equal.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.
> • Easy to understand and implement.
> • Poor in performance as average wait time is high.
> Wait time of each process is following
> Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

> 7.2 Shortest Job First (SJF)
> • Best approach to minimize waiting time.
> • Impossible to implement
> • Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

> 7.3 Priority Based Scheduling
> • Each process is assigned a priority. Process with highest priority is to be executed first and so on.
> • Processes with same priority are executed on first come first serve basis.

• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Question sources

Computer Science MCQ practice collection · Question 695

Question 169 | AM0696

Which algorithm gives priority to the tasks with the shortest burst time?

A. First-Come, First-Served (FCFS)

B. Round Robin

C. Shortest Job Next (SJN)

D. Priority Scheduling

Hint

Compare the ready jobs using the stated burst or remaining-time rule.

Answer C: Shortest Job Next (SJN)

SJF on PDF p23 requires knowing how long a process will take and favors shortest jobs. C uses the synonymous SJN label. Priority can incorporate time requirements, but does not necessarily rank by shortest burst.

Why each choice is right or wrong

A - Incorrect: FCFS chooses order of arrival (p23).

B - Incorrect: RR chooses fixed turns (p25).

C - Correct: Shortest-job selection uses the known execution duration (p23).

D - Incorrect: Assigned priorities can reflect many requirements, not necessarily shortest burst (p24).

Primary note proof

Study-note section OS16

Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS15

FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS18

RR serves a ready queue cyclically for at most quantum q per turn.

Study-note section OS17

Priority scheduling chooses the highest-priority ready job.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science MCQ practice collection · Question 696

Question 170 | AM0698

Which scheduling algorithm is most suitable for time-sharing systems?

A. First-Come, First-Served (FCFS)

B. Round Robin

C. Shortest Job Next (SJN)

D. Priority Scheduling

Hint

Compare interaction, CPU sharing and deadline requirements under the stated system model.

Answer B: Round Robin

PDF p7 describes time-sharing as short bursts/quantums for many users, seeking fast response. RR's fixed quantum and preemption (p25) directly implement that pattern, making B the matching policy.

Why each choice is right or wrong

A - Incorrect: A long early FCFS job can delay other users, unlike repeated short turns.

B - Correct: Fixed quantum sharing directly implements p7's time-sharing pattern.

C - Incorrect: SJN requires knowing lengths, not a periodic-turn guarantee.

D - Incorrect: Priority ordering alone does not guarantee each user a quantum.

Primary note proof

Study-note section OS02

> Time-sharing uses rapid scheduling/preemption to support interactive users.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 7 | lines 20-33

> Time sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multi-programmed Batch Systems and Time-Sharing Systems is that in case of multi-programmed batch systems, objective is to maximize processor use, whereas in Time-Sharing Systems objective is to minimize response time.
> Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction

processing, processor execute each user program in a short burst or quantum of computation. That is if n users are present, each user can get time quantum. When the user submits the command, the response time is in few seconds at most.
Operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science MCQ practice collection · Question 698

Question 171 | AM0699

Which correctly implemented policies can still cause starvation under an ongoing stream of preferred arrivals? Assume finite bursts, ordinary RR with tail insertion, FCFS without bypass, non-preemptive SJN, and fixed priority without aging. Select all that apply.

A. First-Come, First-Served (FCFS)

B. Round Robin

C. Shortest Job Next (SJN)

D. Priority Scheduling

Hint

Check priority direction, preemption and whether waiting can change effective rank.

Answer C, D: Shortest Job Next (SJN); Priority Scheduling

Starvation can arise from the policy even when its implementation is correct. SJN repeatedly favors short arrivals over an older long burst, while fixed priority can repeatedly favor high-priority arrivals over low-priority work. FCFS preserves arrival order, and RR provides recurring turns under the stated finite-work assumptions. Thus SJN and fixed priority are the two policies with the specified inherent bypass risk.

Why each choice is right or wrong

A - Incorrect: FCFS later arrivals cannot bypass a target with finite predecessor work.

B - Incorrect: Ordinary RR returns peers to recurring turns rather than favoring newcomers indefinitely.

C - Correct: Repeated shorter arrivals can indefinitely postpone a long SJN job.

D - Correct: Continued higher-priority arrivals can indefinitely postpone low-priority work without aging.

Primary note proof

Study-note section OS17

> Fixed-priority scheduling can starve low-priority jobs under continued high-priority load. SJF can also starve long jobs; starvation is not exclusive to fixed-priority scheduling.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt. If the CPU is idle, jump to the next arrival. Long CPU-bound work can make many short jobs wait: the convoy effect. A finite burst with finitely many jobs ahead eventually starts, but waiting can still be large.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science MCQ practice collection · Question 699

Question 172 | AM0701

Which scheduling algorithm has the potential for higher overhead due to frequent context switches?

A. First-Come, First-Served (FCFS)

B. Round Robin

C. Shortest Job Next (SJN)

D. Priority Scheduling

Hint

Trace bounded turns in ready-queue order and include the stated overhead and arrivals.

Answer B: Round Robin

Each RR quantum expiration preempts and saves state (PDF p25). PDF p21 calls context-switch time pure overhead, so frequent short quanta create the overhead risk described.

Why each choice is right or wrong

A - Incorrect: FCFS need not switch after every short quantum.

B - Correct: Repeated quantum expiration creates frequent context saves; p21 identifies their cost.

C - Incorrect: Whole-job SJN does not inherently expire a quantum repeatedly.

D - Incorrect: Priority can preempt frequently under some arrivals, but that is not inherent in its p24 ranking rule.

Primary note proof

Study-note section OS18

> A large quantum approaches FCFS; tiny quanta increase switch overhead.

Study-note section OS05

> A context switch saves one execution context and restores another; useful application work is interrupted by this overhead.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 21 | lines 41-50

> A context switch is the mechanism to store and restore the state or context of a CPU in Process Control block so that a process execution can be resumed from the same point at a later time. Using this technique, a context switcher enables multiple processes to share a single CPU. Context switching is an essential part of a multitasking operating system features.
> When the scheduler switches the CPU from executing one process to execute another, the context switcher saves the content of all processor registers for the process being removed from the CPU, in its process descriptor. The context of a process is represented in the process control block of a process. Context switch time is pure overhead. Context switching can significantly affect performance as modern computers have a lot of general

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Question sources

Computer Science MCQ practice collection · Question 701

Question 173 | AM0702

Two continuously ready user groups have declared CPU entitlements of 3:1, independent of how many tasks each launches. Which listed policy is designed to enforce those unequal per-group shares over time?

A. First-Come, First-Served (FCFS)

B. Round Robin

C. Shortest Job Next (SJN)

D. Fair-Share Scheduling

Hint

Identify the allocation entity, entitlement weights and whether the guarantee is probabilistic.

Answer D: Fair-Share Scheduling

A fair-share policy allocates CPU according to declared weights and participant scope. Weights of 3:1 give intended group shares of 75 and 25 percent. Plain RR supplies equal turns to processes and does not implement those unequal group entitlements independently of task count. FCFS and SJN also lack that share rule. The question asks for the policy design rather than a universal finite-time deadline.

Why each choice is right or wrong

A - Incorrect: FCFS orders bursts by arrival rather than enforcing weighted group service.

B - Incorrect: Plain per-task RR does not enforce unequal group weights independent of task count.

C - Incorrect: SJN selects short bursts rather than proportional group entitlements.

D - Correct: Configured fair-share scheduling implements the declared group-weight allocation.

Primary note proof

Study-note section OS20

> Fair-share scheduling allocates according to stated weights, potentially per user/group rather than per process. Equal per-process RR can let a user obtain more aggregate CPU by launching more processes. Proportional shares address that policy choice; fair need not mean equal.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 25 | lines 10-14

> 7.4 Round Robin Scheduling
> • Each process is provided a fix time to execute called quantum.
> • Once a process is executed for given time period. Process is preempted and other process executes for given time period.
> • Context switching is used to save states of preempted processes.

Operating System module | PDF page 23 | lines 12-17

> 7.1 First Come First Serve (FCFS)
> • Jobs are executed on first come, first serve basis.
> • Easy to understand and implement.
> • Poor in performance as average wait time is high.
> Wait time of each process is following
> Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

> 7.2 Shortest Job First (SJF)
> • Best approach to minimize waiting time.

- Impossible to implement
- Processer should know in advance how much time process will take.

Question sources

Computer Science MCQ practice collection · Question 702

Question 174 | AM0703

Three jobs all arrive at time 0 in listed order P1, P2, P3, with CPU bursts 24, 3, 3 and priorities 2, 1, 3 (smaller number is higher). There is no I/O or overhead. Compare FCFS, RR with quantum 4, non-preemptive SJN, and non-preemptive priority; use listed order for ties. Which has the highest average turnaround for this workload?

A. First-Come, First-Served (FCFS)

B. Round Robin

C. Shortest Job Next (SJN)

D. Priority Scheduling

Hint

Construct a timeline and distinguish completion rates from individual elapsed-time measures.

Answer A: First-Come, First-Served (FCFS)

FCFS completes P1, P2, P3 at 24, 27 and 30, giving average turnaround 27. RR completes them at 30, 7 and 10, averaging 47/3. SJN completes short jobs at 3 and 6, then P1 at 30, averaging 13. Priority runs P2, P1, P3 and averages 20. FCFS is uniquely highest for this specified workload, without a universal worst-policy claim.

Why each choice is right or wrong

A - Correct: FCFS average turnaround is (24+27+30)/3=27, the largest.

B - Incorrect: RR average turnaround is (30+7+10)/3=47/3, below 27.

C - Incorrect: SJN average turnaround is (3+6+30)/3=13.

D - Incorrect: Priority average turnaround is (3+27+30)/3=20.

Primary note proof

Study-note section OS13

For arrival A, first start S and completion C, response=S−A and turnaround=C−A. For a single CPU burst B with no I/O, waiting=C−A−B.

Study-note section OS15

FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS16

Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs.

Study-note section OS17

Priority scheduling chooses the highest-priority ready job. Check whether a smaller or larger numeric value means higher priority. State whether a high-priority arrival preempts.

Study-note section OS18

RR serves a ready queue cyclically for at most quantum q per turn. Unfinished jobs return to the tail; completed/blocked jobs leave.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 23 | lines 12-17

7.1 First Come First Serve (FCFS)
• Jobs are executed on first come, first serve basis.
• Easy to understand and implement.
• Poor in performance as average wait time is high.
Wait time of each process is following
Process Wait Time: Service Time - Arrival Time

Operating System module | PDF page 23 | lines 23-26

7.2 Shortest Job First (SJF)
• Best approach to minimize waiting time.
• Impossible to implement
• Processer should know in advance how much time process will take.

Operating System module | PDF page 25 | lines 10-14

7.4 Round Robin Scheduling
• Each process is provided a fix time to execute called quantum.
• Once a process is executed for given time period. Process is preempted and other process executes for given time period.
• Context switching is used to save states of preempted processes.

Operating System module | PDF page 24 | lines 10-15

7.3 Priority Based Scheduling
• Each process is assigned a priority. Process with highest priority is to be executed first and so on.
• Processes with same priority are executed on first come first serve basis.
• Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Operating System module | PDF page 25 | lines 15-21

Wait time of each process is following
Process Wait Time: Service Time - Arrival Time
P0 (0-0) + (12-3) = 9

P1 (3-1) = 2
P2 6-2) + (15-9) = 10
P3 (9-3) + (18-12) = 12
Average Wait Time: (9+2+10+12) / 4 = 8.25

Operating System module | PDF page 23 | lines 16-22

Wait time of each process is following
Process Wait Time: Service Time - Arrival Time
P0 0 - 0 = 0
P1 5 - 1 = 4
P2 8 - 2 = 6
P3 16 - 3 = 13
Average Wait Time: (0+4+6+13)/4 = 5.75

Question sources

Computer Science MCQ practice collection · Question 703

Question 175 | CS0218

Which one of the following is a voluntary termination of a process?

A. Error exit

B. Killed by another process

C. Terminated by another process

D. Fatal exit

Hint

Separate creation, image replacement and termination; classify termination by its cause.

Answer A: Error exit, when the program detects an error and chooses to terminate.

OS07 explicitly classifies program-detected error exits as voluntary. Being killed or terminated by another process is imposed, and a fatal unrecoverable fault is involuntary. The distinction is the termination decision's origin, not whether the run succeeded: a program can voluntarily exit because its input is invalid or it detects another handled error.

Why each choice is right or wrong

A - Correct: A program-detected error exit is a deliberate voluntary termination path, even though execution was unsuccessful.

B - Incorrect: Being killed by another process is imposed externally and is therefore involuntary.

C - Incorrect: Termination by another process is external rather than a voluntary exit chosen by the program.

D - Incorrect: A fatal unrecoverable fault causes involuntary termination rather than a deliberate program error exit.

Primary note proof

Study-note section OS07

Normal completion and a program-detected error exit are voluntary; an unrecoverable fault or being killed is involuntary. exit() terminates normally through the runtime; abort() requests abnormal termination.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 16 | lines 29-39

1. New The process is being created.
2.
Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
3.
Running Process instructions are being executed (i.e. The process that is currently being executed).
4.
Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
5. Terminated The process has finished execution.

Operating System module | PDF page 19 | lines 20-25

Non-Running Processes that are not running are kept in queue, waiting for their turn to execute. Each entry in the queue is a pointer to a particular process. Queue is implemented by using linked list. Use of dispatcher is as follows. When a process is interrupted, that process is transferred in the waiting queue. If the process has completed or aborted, the process is discarded. In either case, the dispatcher then selects a process from the queue to execute.

Operating System module | PDF page 39 | lines 5-11

Check an internal table for this process, to determine whether the reference was a valid or it was an invalid memory access.
Step 2
If the reference was invalid, terminate the process. If it was valid, but page have not yet brought in, page in the latter.
Step 3 Find a free frame.
Step 4 Schedule a disk operation to read the desired page into the newly allocated frame.

Question sources

Computer Science Exit Exam practice collection · Question 218

Question 176 | CS0219

Which statements are false in a system with one-to-one kernel-supported threads? In A, “children” means a process may create another process and a running thread may invoke a thread-creation operation; it does not require an OS thread-parent hierarchy. In B, “execute sequentially” means threads are required to finish one after another without interleaving. Select all that apply.

A. Both can create children

B. Threads within a process execute sequentially

C. Processes are not independent of one another

D. If one thread is blocked, another thread can run

Hint

Compare per-thread execution context with shared process resources and the stated thread mapping.

Answer B, C: Threads within a process execute sequentially; Processes are not independent of one another

Threads within one process need not finish sequentially; their execution may interleave or run on multiple CPUs. Independent processes can also exist, so C’s universal dependence claim is false. The defined creation operations make A true without imposing a thread-parent hierarchy. Under one-to-one mapping, one blocked thread need not block every other thread, making D a valid possibility rather than a false statement.

Why each choice is right or wrong

A - Incorrect: The defined process and thread creation operations are allowed, so A is true.

B - Correct: Mandatory finish-before-next sequential execution is false; threads can interleave or run in parallel.

C - Correct: Universal process dependence is false because independent processes can exist.

D - Incorrect: One-to-one mapping permits another runnable thread to execute when one blocks, so D is true.

Primary note proof

Study-note section OS08

> A thread has its own PC, registers and stack. Threads in a process share code, address space and usually open-file resources. A process can contain many threads; a thread belongs to one process at a time. Threads are not a group of processes and need not execute sequentially. Independent processes can exist; cooperating ones communicate/share data.

Study-note section OS08

> One-to-one permits another kernel thread to run when one blocks.

Study-note section OS07

Processes arise during initialization, user requests, service requests or a running process's creation call.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 27 | lines 6-14

A thread is a flow of execution through the process code, with its own program counter, system registers and stack. A thread is also called a light weight process. Threads provide a way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.
Each thread belongs to exactly one process and no thread can exist outside a process. Each thread represents a separate flow of control. Threads have been successfully used in implementing network servers and web server. They also provide a suitable foundation for parallel execution of applications on shared memory multiprocessors. Following figure shows

Operating System module | PDF page 28 | lines 4-13

In multiple processes each process operates
independently of the others.
One thread can read, write or change
another thread's data.
8.2 Advantages of Thread
• Thread minimizes context switching time.
• Use of threads provides concurrency within a process.
• Efficient communication.
• Economy- It is more economical to create and context switch threads.
• Utilization of multiprocessor architectures to a greater scale and efficiency.

Operating System module | PDF page 29 | lines 4-16

In this case, thread management done by the Kernel. There is no thread management code in the application area. Kernel threads are supported directly by the operating system. Any application can be programmed to be multithreaded. All of the threads within an application are supported within a single process.
The Kernel maintains context information for the process as a whole and for individuals' threads within the process. Scheduling by the Kernel is done on a thread basis. The Kernel performs thread creation, scheduling and management in Kernel space. Kernel threads are generally slower to create and manage than the user threads.
Advantages
• Kernel can simultaneously schedule multiple threads from the same process on multiple processes.
• If one thread in a process is blocked, the Kernel can schedule another thread of the same process.

Operating System module | PDF page 30 | lines 4-7

Many to one model maps many user level threads to one Kernel level thread. Thread management is done in user space. When thread makes a blocking system call, the entire

process will be blocks. Only one thread can access the Kernel at a time, so multiple threads are unable to run in parallel on multiprocessors.

Operating System module | PDF page 19 | lines 6-12

A newly arrived process is put in the ready queue. Processes waits in ready queue for allocating the CPU. Once the CPU is assigned to a process, then that process will execute. While executing the process, any one of the following events can occur.
• The process could issue an I/O request and then it would be placed in an I/O queue.
• The process could create new sub process and will wait for its termination.
• The process could be removed forcibly from the CPU, as a result of interrupt and put back in the ready queue.

Operating System module | PDF page 28 | lines 20-29

In this case, application manages thread management kernel is not aware of the existence of threads. The thread library contains code for creating and destroying threads, for passing message and data between threads, for scheduling thread execution and for saving and restoring thread contexts. The application begins with a single thread and begins running in that thread.
Advantages
• Thread switching does not require Kernel mode privileges.
• User level thread can run on any operating system.
• Scheduling can be application specific in the user level thread.
• User level threads are fast to create and manage.

Question sources

Computer Science Exit Exam practice collection · Question 219

Question 177 | CS0220

Which statements are false under a classification based on simultaneous users? A single user may run multiple applications, and B means concurrent database accesses by those applications, not by additional users. In C, “at most one” describes the system’s supported capacity. Select all that apply.

A. In single user systems many users can login simultaneously

B. Single user systems can allow concurrent access to the database

C. Multiuser system can allow at most one user at a time

D. In Multiuser system many users can access the system concurrently

Hint

Compare interaction, CPU sharing and deadline requirements under the stated system model.

Answer A, C: In single user systems many users can login simultaneously; Multiuser system can allow at most one user at a time

Single-user and multiuser describe concurrent user access rather than application count. A therefore contradicts single-user status, and C contradicts the supported multiple-user capacity. One user may still run several applications accessing a database concurrently, which is the explicit meaning of B here. D states the multiuser capability correctly. The selected false statements are A and C.

Why each choice is right or wrong

A - Correct: Many simultaneous logged-in users contradict the single-user definition.

B - Incorrect: Concurrent applications belonging to one user do not violate single-user status, so B is true.

C - Correct: A capacity of at most one user contradicts the multiuser definition.

D - Incorrect: Supporting concurrent access by many users is the multiuser capability, so D is true.

Primary note proof

Study-note section OS02

> Single-user and multiuser describe concurrent user access, not the number of applications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 54 | lines 5-17

> Portable - Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.
> Open Source - Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
> Multi-User - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
> Multiprogramming - Linux is a multiprogramming system means multiple applications can run at same time.
> Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.
> Shell - Linux provides a special interpreter program which can be used to execute

Operating System module | PDF page 7 | lines 20-33

> Time sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multi-programmed Batch Systems and Time-Sharing Systems is that in case of multi-programmed batch systems, objective is to maximize processor use, whereas in Time-Sharing Systems objective

is to minimize response time.
Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, processor execute each user program in a short burst or quantum of computation. That is if n users are present, each user can get time quantum. When the user submits the command, the response time is in few seconds at most.
Operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems

Question sources

Computer Science Exit Exam practice collection · Question 220

Question 178 | CS0221

When a process executes code that possibly changes the value of shared data, its execution has to be mutually exclusive. Therefore, the process is in:

A. Deadlock

B. Terminated

C. Critical section

D. Blocked

Hint

Identify conflicting shared accesses and the safety or progress property required.

Answer C: Critical section.

OS10 defines a critical section as code accessing shared state that needs protection. The stem describes execution that may change shared data and therefore requires mutual exclusion, directly matching that definition. Deadlock and blocking are waiting states or failures, while termination means execution has ended. None identifies the protected shared-state code currently being executed.

Why each choice is right or wrong

A - Incorrect: Deadlock involves blocked dependencies, whereas the stem describes protected code actively accessing shared data.

B - Incorrect: A terminated process has ended execution, contradicting the stem’s currently executing shared-data operation.

C - Correct: Code that accesses shared data and requires mutual exclusion is a critical section.

D - Incorrect: A blocked process cannot proceed pending an event; the stem instead identifies the code requiring protection.

Primary note proof

Study-note section OS10

A race occurs when results depend on uncoordinated interleaving of conflicting accesses. A critical section is code accessing shared state that needs protection, not the resource itself. A correct exclusion solution requires mutual exclusion, progress and bounded waiting under stated assumptions.

Study-note section OS06

New processes are created/admitted; ready means eligible for CPU but not currently running. Running uses a CPU. Waiting/blocked means unable to proceed until an event; terminated means execution has ended.

Study-note section OS22

A set is deadlocked when each member waits for an event/resource that only another blocked member can provide.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Operating System module | PDF page 10 | lines 23-28

• Loads a program into memory.
• Executes the program.
• Handles program's execution.
• Provides a mechanism for process synchronization.
• Provides a mechanism for process communication.
• Provides a mechanism for deadlock handling.

Operating System module | PDF page 16 | lines 29-39

1. New The process is being created.
2.
Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run.
3.
Running Process instructions are being executed (i.e. The process that is currently being executed).
4.
Waiting The process is waiting for some event to occur (such as the completion of an I/O operation).
5. Terminated The process has finished execution.

Question sources

Computer Science Exit Exam practice collection · Question 221

Question 179 | NS0308

Which one of the following function of operating system is categorized under process management?

A. Allocates the device in the efficient way

B. De-allocates memory when a process releases it.

C. Allocates the memory when the process requests it to do so.

D. Allocates the processor (CPU) to a process

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer D: Allocate the CPU to a process.

Allocating the CPU to a ready process is the scheduling role within process management. Device allocation belongs to device management. Allocating requested memory and reclaiming memory that has been released belong to memory management. Specifying the resource in the deallocation option removes its former overlap with several managers and leaves processor allocation as the distinct process-management function.

Why each choice is right or wrong

A - Incorrect: Allocating a device is a device-management responsibility.

B - Incorrect: Reclaiming released memory is a memory-management responsibility under the stated resource classification.

C - Incorrect: Allocating requested memory belongs to memory management.

D - Correct: Selecting a ready process to receive the CPU is the process-scheduling responsibility.

Primary note proof

Study-note section OS06

> Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work. The dispatcher performs the handoff/context restore after selection.

Study-note section OS-S03

> Memory management tracks which primary-memory regions are free or in use, assigns memory to processes and data, and deallocates it when no longer needed or when a process terminates. Allocation and deallocation concern memory ownership; CPU selection belongs to scheduling.

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs.

Question sources

Computer Science Exit Exam · 2018 · Question 29

Question 180 | NS0316

What is the primary function of an operating system?

A. To connect to the internet

B. To create documents and spreadsheets

C. To design computer hardware

D. To act as an intermediary between users and computer hardware

Hint

Separate broad resource-management goals from individual services or applications.

Answer D: Act as the service/resource intermediary between users/programs and hardware.

OS01 describes the OS allocating hardware resources and providing program services, with convenient interfaces and abstractions. D captures that intermediary role. Internet connectivity and document creation are narrower tasks, and hardware design is not a runtime OS service. The answer combines convenience and resource control rather than reducing the OS to one application or device function.

Why each choice is right or wrong

A - Incorrect: Internet access is one supported communication activity rather than the general purpose of an operating system.

B - Incorrect: Document and spreadsheet creation are application tasks that use OS services rather than define the OS.

C - Incorrect: Designing hardware is not the runtime resource/service function described for an operating system.

D - Correct: The OS provides interfaces and controlled resource services linking users/programs to the underlying hardware.

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process. Application development tools use OS services but application development itself is not a core resource-management function.

Question sources

Computer Science Exit Exam · 2018 · Question 41

Question 181 | NS0323

A user installs a new printer. The OS automatically detects and configures it. This is an example of:

A. Plug-and-play functionality

B. Manual driver installation

C. Manual hardware management

D. Resource allocation

Hint

Separate device hardware, device-specific software and general event or transfer mechanisms.

Answer A: Plug-and-play functionality.

OS34 defines plug-and-play as discovering/configuring compatible new hardware, which directly matches automatic printer setup. A driver may still be needed, so automatic configuration does not imply driver-free operation. Manual driver installation and hardware management contradict the scenario's automatic action. Resource allocation may occur during setup but is broader than the specific feature named.

Why each choice is right or wrong

A - Correct: Automatic detection and configuration of the printer is the note's plug-and-play behavior.

B - Incorrect: The stem specifies automatic setup, not user-performed manual installation of a driver.

C - Incorrect: Manual hardware management does not match the automatic discovery/configuration described in the scenario.

D - Incorrect: Allocation can be part of setup, but it does not specifically name automatic device discovery/configuration.

Primary note proof

Study-note section OS34

> Plug-and-play discovers/configures compatible new hardware but may still require a driver.

Question sources

Computer Science Exit Exam · 2018 · Question 48

Question 182 | NS0330

In a system, resources cannot be forcibly taken from processes; they must be released voluntarily. Which deadlock condition does this represent?

A. No Preemption

B. Hold and Wait

C. Mutual Exclusion

D. Circular Wait

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer A: No preemption.

OS22 defines no resource preemption as voluntary release by the current holder. The stem repeats that resources cannot be forcibly taken, giving a direct match. Holding while requesting another resource, one-at-a-time ownership and a closed chain of waits are separate conditions and do not by themselves express the withdrawal restriction.

Why each choice is right or wrong

A - Correct: The holder must release voluntarily, so the system cannot forcibly preempt the resource.

B - Incorrect: Hold-and-wait concerns retaining one resource while requesting another rather than whether withdrawal is allowed.

C - Incorrect: Mutual exclusion limits simultaneous ownership, not whether allocated resources can be forcibly reclaimed.

D - Incorrect: Circular wait concerns a closed dependency chain, not the resource-release rule specified here.

Primary note proof

Study-note section OS22

> Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Question sources

Computer Science Exit Exam · 2018 · Question 57

Question 183 | NS0359

How does an operating system improve efficiency in a multi-user environment?

A. By allowing only one user to access the system at a time

B. By disabling all background services

C. By using time-sharing and process scheduling to allocate CPU time fairly among users and processes

D. By requiring manual intervention for each task

Hint

Compare interaction, CPU sharing and deadline requirements under the stated system model.

Answer C: Time-sharing and scheduling distribute CPU service among users/processes.

OS02 describes time-sharing as rapid scheduling/preemption supporting interactive users. OS20 explains that fair allocation needs a share policy and may operate per user/group. C describes those relevant mechanisms rather than disabling services or requiring manual intervention. Its fairly should be understood under a specified policy, not as an automatic equality guarantee from time-sharing alone.

Why each choice is right or wrong

A - Incorrect: Restricting access to one user contradicts the multiuser setting rather than explaining its efficient sharing.

B - Incorrect: Disabling every background service is neither required nor the time-sharing/resource-allocation mechanism described.

C - Correct: Scheduling/time-sharing interleave useful work and can allocate service according to the system's fairness policy.

D - Incorrect: Manual intervention for each task adds administration rather than explaining efficient automatic CPU allocation.

Primary note proof

Study-note section OS02

> Multiprogramming keeps several jobs available so another can use the CPU while one waits; it does not require fixed time slices. Time-sharing uses rapid scheduling/preemption to support interactive users. A single CPU interleaves execution; multiple CPUs can execute work truly in parallel.

Study-note section OS20

> Fair-share scheduling allocates according to stated weights, potentially per user/group rather than per process. Equal per-process RR can let a user obtain more aggregate CPU by launching more processes. Proportional shares address that policy choice; fair need not mean equal.

Question sources

Computer Science Exit Exam · 2018 · Question 97

Question 184 | NS0684

What is the primary role of an Operating System?

A. Run application software only

B. Manage hardware and software resources and act as interface between user and hardware

C. Provide internet connectivity

D. Store user files permanently

Hint

Separate broad resource-management goals from individual services or applications.

Answer B: Manage resources and provide interfaces/services.

OS01 defines both allocation of CPU/memory/storage/devices/communications and services to programs, while describing convenient interfaces. B combines those core responsibilities. Running applications only, providing connectivity only or permanently storing files each captures a narrower task and omits the overall system-management role. The primary role is therefore the resource/interface combination, not a single service.

Why each choice is right or wrong

A - Incorrect: Application execution is supported, but only excludes the OS's other essential resource and service duties.

B - Correct: Resource management plus user/program interfaces matches the combined OS role defined here.

C - Incorrect: Internet connectivity is one communication service rather than the complete primary OS role.

D - Incorrect: Permanent file storage is a storage-device/filesystem service and not the complete operating-system purpose.

Primary note proof

Study-note section OS01

> An OS allocates CPU time, memory, storage, devices and communication resources, and provides services to programs. Convenience means making tasks easy through consistent interfaces, files and device abstractions. Efficiency means using resources effectively; it is not synonymous with the fastest response for every process. Application development tools use OS services but application development itself is not a core resource-management function.

Question sources

Operating Systems review questions · Question 1

Question 185 | NS0685

Which type of OS is designed to process large numbers of similar jobs without user interaction during execution?

A. Real-Time OS

B. Time-Sharing OS

C. Batch System

D. Single-User OS

Hint

Compare interaction, CPU sharing and deadline requirements under the stated system model.

Answer C: Batch system.

OS02 defines batch jobs as running with little interaction after submission. That matches processing submitted similar jobs without user interaction during execution. Time-sharing targets interactive users, real-time targets timing correctness, and single-user classifies concurrent access rather than job interaction. The job-processing clue therefore selects batch rather than a user-count or deadline category.

Why each choice is right or wrong

A - Incorrect: Real-time classification concerns meeting deadlines rather than the absence of user interaction during submitted job processing.

B - Incorrect: Time-sharing supports interactive users through rapid scheduling, unlike the noninteractive batch description.

C - Correct: Batch processing runs submitted jobs with little interaction, directly matching the scenario.

D - Incorrect: Single-user describes concurrent user access, not whether a workload is submitted and processed as batches.

Primary note proof

Study-note section OS02

> Batch jobs run with little interaction after submission. Multiprogramming keeps several jobs available so another can use the CPU while one waits; it does not require fixed time slices. Time-sharing uses rapid scheduling/preemption to support interactive users.

Study-note section OS02

> Single-user and multiuser describe concurrent user access, not the number of applications. Network systems expose remote services; distributed systems coordinate work across machines, sometimes hiding location. Real-time correctness includes timing: hard deadlines must be met, while soft deadline misses degrade service.

Question sources

Operating Systems review questions · Question 2

Question 186 | NS0686

In a Process Control Block (PCB), what does the Program Counter store?

A. The process priority level

B. The address of the next instruction to be executed for that process

C. The amount of CPU time used

D. The list of open files

Hint

Distinguish stored instructions, an executing instance and its saved execution state.

Answer B: The saved next-instruction position for resuming the process.

OS05 says the PCB records PC/registers alongside scheduling, memory/accounting and I/O information, and a saved PC identifies where execution resumes. Thus B describes its specific field role. Priority, CPU usage and open files can be other PCB information, but they are not the program-counter value. The question requires distinguishing fields within one record rather than excluding them from the PCB entirely.

Why each choice is right or wrong

A - Incorrect: Priority is scheduling information in the PCB, not the saved next-instruction position held by PC.

B - Correct: The saved PC records the instruction position at which the process will resume execution.

C - Incorrect: CPU time consumed is accounting information rather than the program-counter address.

D - Incorrect: Open files belong to I/O/open-resource state, not to the saved instruction-pointer field.

Primary note proof

Study-note section OS05

> A saved PC identifies where execution resumes.

Question sources

Operating Systems review questions · Question 3

Question 187 | NS0687

Which process state indicates that a process is waiting for an I/O event to complete?

A. Ready

B. Running

C. Waiting

D. Terminated

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer C: Waiting/blocked.

OS06 defines waiting/blocked as unable to proceed until an event and maps blocking I/O from running to waiting. Therefore a process waiting for I/O completion is in Waiting. It is not ready while the event prevents progress, not running during the wait, and not terminated merely because its request is outstanding. Completion normally makes it ready afterward.

Why each choice is right or wrong

A - Incorrect: Ready means eligible for CPU immediately; the outstanding blocking I/O prevents that eligibility.

B - Incorrect: Running means currently using a CPU, unlike the blocked interval awaiting I/O completion.

C - Correct: Waiting/blocked is the state for a process that cannot proceed until its I/O event completes.

D - Incorrect: Terminated means execution has ended, not temporarily paused pending completion of an I/O request.

Primary note proof

Study-note section OS06

> New processes are created/admitted; ready means eligible for CPU but not currently running. Running uses a CPU. Waiting/blocked means unable to proceed until an event; terminated means execution has ended. Running→ready follows time-slice expiry or another preemption. Running→waiting follows blocking I/O. I/O completion normally gives waiting→ready, not a guaranteed immediate running state.

Question sources

Operating Systems review questions · Question 4

Question 188 | NS0688

What is the primary purpose of a Context Switch?

A. To permanently stop a process

B. To save the state of the current process and restore the state of the next scheduled process

C. To allocate more memory to a process

D. To terminate a low-priority process

Hint

Distinguish stored instructions, an executing instance and its saved execution state.

Answer B: Save one execution context and restore another.

OS05 directly defines a context switch as saving one execution context and restoring another, with overhead interrupting useful application work. B matches. A process may later resume, so switching does not permanently stop or kill it. Allocating memory is another management operation, and a privilege transition alone need not switch processes.

Why each choice is right or wrong

A - Incorrect: The outgoing process can resume later; permanent stopping is termination rather than context switching.

B - Correct: Saving current state and restoring the selected execution context is the defined context-switch purpose.

C - Incorrect: Memory allocation changes resource assignment, not the saved/restored execution context defining a switch.

D - Incorrect: A context switch need not terminate any process, regardless of priority.

Primary note proof

Study-note section OS05

> A context switch saves one execution context and restores another; useful application work is interrupted by this overhead. An interrupt or system call can switch privilege without switching process.

Question sources

Operating Systems review questions · Question 5

Question 189 | NS0689

Which scheduler is responsible for selecting which process from the ready queue should be executed next on the CPU?

A. Long-Term Scheduler

B. Medium-Term Scheduler

C. Short-Term (CPU) Scheduler

D. I/O Scheduler

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer C: Short-term (CPU) scheduler.

OS06 assigns admission to the long-term scheduler, suspension/swapping to the medium-term scheduler and ready-work selection to the short-term scheduler. The question specifically asks which ready process receives the CPU next, so C is exact. The dispatcher then performs the handoff; that is distinct from selecting the ready process. Device request ordering is not CPU ready-queue selection.

Why each choice is right or wrong

A - Incorrect: The long-term scheduler admits jobs rather than making each immediate ready-queue CPU selection.

B - Incorrect: The medium-term scheduler suspends/swaps work rather than choosing the next CPU-ready process.

C - Correct: The short-term CPU scheduler selects the next eligible process from ready work.

D - Incorrect: An I/O scheduler orders device work rather than selecting the next process for the CPU.

Primary note proof

Study-note section OS06

> The ready queue contains eligible work; device queues hold blocked requests. The broad textbook job queue contains processes in the system, while an admission queue more narrowly holds jobs awaiting memory/admission. Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work. The dispatcher performs the handoff/context restore after selection.

Question sources

Operating Systems review questions · Question 6

Question 190 | NS0690

What is the role of the Medium-Term Scheduler?

A. Decide which jobs enter the system

B. Pick the next process for the CPU

C. Swap processes between main memory and secondary storage to manage memory

D. Handle device driver requests

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer C: Suspend/swap processes between resident and backing-store states.

OS06 explicitly assigns suspension/swapping to the medium-term scheduler. OS27 distinguishes traditional whole-process swapping to backing storage from page movement. Thus C is the matching memory/load-control role. Admitting jobs belongs to long-term scheduling, picking the next ready process to short-term scheduling, and servicing drivers to device-management paths.

Why each choice is right or wrong

A - Incorrect: Deciding admission into the system is the long-term scheduler's role in the supplied model.

B - Incorrect: Selecting the next CPU process belongs to the short-term scheduler, not the medium-term one.

C - Correct: Medium-term scheduling suspends/swaps processes to control resident load and memory use.

D - Incorrect: Device-driver requests concern I/O service rather than the scheduler's suspension/swapping function.

Primary note proof

Study-note section OS06

> Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Study-note section OS27

> Swapping traditionally moves a whole process to backing store; paging moves individual pages.

Study-note section OS34

> A driver converts OS requests into device-specific operations.

Question sources

Operating Systems review questions · Question 7

Question 191 | NS0691

A Race Condition occurs when:

A. Two processes finish execution at the same time

B. Multiple processes access shared resources simultaneously without proper synchronization, causing unpredictable results

C. A high-priority process always runs before a low-priority one

D. A process is preempted by the OS scheduler

Hint

Identify conflicting shared accesses and the safety or progress property required.

Answer B: Uncoordinated conflicting shared accesses make results depend on interleaving.

OS10 defines a race by results depending on uncoordinated interleaving of conflicting accesses. B is the intended match, with shared access understood to include a conflict such as an update; simultaneous read-only access alone is insufficient. Coincident completion, ordinary priority selection and preemption do not by themselves establish a shared-state race.

Why each choice is right or wrong

A - Incorrect: Two completions at the same time need not involve conflicting shared accesses or interleaving-dependent results.

B - Correct: Unsynchronized conflicting shared accesses can make results depend on execution interleaving, which is the race condition.

C - Incorrect: Priority order alone is a scheduling choice and does not establish a conflicting shared-data race.

D - Incorrect: Preemption can expose a race but is not itself sufficient without conflicting unprotected accesses.

Primary note proof

Study-note section OS10

> A race occurs when results depend on uncoordinated interleaving of conflicting accesses.

Study-note section OS10

> Two threads read counter=0, each computes 1 and each stores 1. Two increments produce 1 instead of 2. Protecting the entire read-modify-write with one lock prevents that lost update.

Question sources

Operating Systems review questions · Question 8

Question 192 | NS0692

Which of the following is NOT a necessary condition for deadlock?

A. Mutual Exclusion

B. Hold and Wait

C. Preemption

D. Circular Wait

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer C: Preemption is not a necessary condition; no-preemption is.

OS22 names mutual exclusion, hold-and-wait, no resource preemption and circular wait as the necessary set. The option Preemption reverses one of those conditions. Therefore C is the exception, while the other three are genuine members. This resource-preemption rule is separate from whether CPU scheduling uses timer preemption.

Why each choice is right or wrong

A - Incorrect: Mutual exclusion is explicitly one of the four necessary conditions in the standard resource model.

B - Incorrect: Hold-and-wait is explicitly necessary: a process retains a resource while requesting another.

C - Correct: The required condition is no-preemption; permitting preemption is not one of the necessary conditions.

D - Incorrect: Circular wait is explicitly a required closed dependency chain in the standard model.

Primary note proof

Study-note section OS22

> Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Study-note section OS22

> All four are necessary in the standard reusable-resource model.

Question sources

Operating Systems review questions · Question 9

Question 193 | NS0693

All three processes arrive at time 0 and are ready immediately. Their CPU burst times are P1 = 24 ms, P2 = 3 ms and P3 = 3 ms. Using non-preemptive FCFS with tie order P1, P2, P3, and ignoring context-switch overhead, what is the average waiting time?

A. 3 ms

B. 9 ms

C. 17 ms

D. 27 ms

Hint

Follow arrival order, include idle periods and distinguish waiting from completion time.

Answer C: 17 ms.

FCFS runs P1 from 0 to 24 ms, P2 from 24 to 27 ms, and P3 from 27 to 30 ms. Because every process arrives at 0, their waiting times are 0, 24 and 27 ms. The average is (0+24+27)/3 = 51/3 = 17 ms, so C is correct. The average turnaround is a different quantity: (24+27+30)/3 = 27 ms.

Why each choice is right or wrong

A - Incorrect: 3 ms is the burst time of P2 or P3, not the average ready-queue wait.

B - Incorrect: The three waits total 51 ms, so dividing by three cannot give 9 ms.

C - Correct: The waits 0, 24 and 27 ms average to 17 ms.

D - Incorrect: 27 ms is the average turnaround time in this example, not the average waiting time.

Primary note proof

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS15

> All ready at time 0, tie order P1=24,P2=3,P3=3: schedule 0-24 P1,24-27 P2,27-30 P3. Waiting times 0,24,27 average to 17. Turnarounds 24,27,30 average to 27. If arrivals are not simultaneous, recompute using their actual times.

Study-note section OS13

> For arrival A, first start S and completion C, response=S−A and turnaround=C−A. For a single CPU burst B with no I/O, waiting=C−A−B.

Question sources

Operating Systems review questions · Question 10

Question 194 | NS0694

What problem does the 'Aging' technique solve in Priority Scheduling?

A. Deadlock

B. Starvation of low-priority processes

C. Race conditions

D. Memory fragmentation

Hint

Check priority direction, preemption and whether waiting can change effective rank.

Answer B: Starvation of low-priority processes.

OS17 defines aging as increasing waiting jobs' effective priority to reduce indefinite postponement under suitable rules/workloads. Thus B names the problem it addresses. It does not remove a deadlock dependency cycle, coordinate conflicting memory updates or consolidate fragmented memory. The result still depends on the priority/aging policy; naming aging alone does not provide an unconditional deadline guarantee.

Why each choice is right or wrong

A - Incorrect: Aging changes priority rather than breaking the blocked resource dependencies of a deadlock.

B - Correct: Increasing effective priority of waiting low-priority jobs addresses their starvation under sustained higher-priority load.

C - Incorrect: Races require shared-access synchronization; priority aging is not a mutual-exclusion mechanism.

D - Incorrect: Memory fragmentation concerns allocation layout rather than the priority of waiting CPU jobs.

Primary note proof

Study-note section OS17

> Aging increases waiting jobs' effective priority according to a rule, reducing indefinite postponement when the priority scheme and workload permit. It is not a deadlock cure.

Study-note section OS10

> A race occurs when results depend on uncoordinated interleaving of conflicting accesses.

Study-note section OS29

> Internal fragmentation is unused space inside an allocated unit. External fragmentation is sufficient total free space split into holes that cannot satisfy a required contiguous allocation.

Question sources

Operating Systems review questions · Question 11

Question 195 | NS0695

Which scheduling algorithm is known as preemptive SJF?

A. Round Robin

B. FCFS

C. Shortest Remaining Time First (SRTF)

D. Priority Scheduling

Hint

Compare the ready jobs using the stated burst or remaining-time rule.

Answer C: Shortest Remaining Time First.

OS16 distinguishes non-preemptive SJF/SJN from SRTF, which reconsiders at arrivals and can preempt for shorter remaining work. Therefore SRTF is the preemptive SJF variant. RR uses fixed cyclic turns, FCFS arrival order and generic priority an independently defined ranking. The original burst length and the remaining length must not be confused after a job has executed.

Why each choice is right or wrong

A - Incorrect: RR selects cyclic quantum turns rather than the shortest remaining CPU work.

B - Incorrect: FCFS follows arrival order and is ordinarily non-preemptive rather than preemptive shortest-job scheduling.

C - Correct: SRTF chooses minimum remaining work and can interrupt a running job for a shorter arrival.

D - Incorrect: Generic priority scheduling does not specifically rank by remaining burst length and can be either variant.

Primary note proof

Study-note section OS16

> Non-preemptive SJF/SJN chooses the shortest next burst among currently ready jobs. SRTF reconsiders on arrivals and selects the shortest remaining work; a shorter newcomer can preempt. Future burst lengths normally need estimates.

Study-note section OS18

> RR serves a ready queue cyclically for at most quantum q per turn.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Question sources

Operating Systems review questions · Question 12

Question 196 | NS0696

In Round Robin scheduling, what is the key challenge?

A. Managing priority levels

B. Determining the appropriate time quantum length

C. Avoiding deadlocks

D. Allocating memory to processes

Hint

Trace bounded turns in ready-queue order and include the stated overhead and arrivals.

Answer B: Choose an appropriate time quantum.

OS18 says a large RR quantum approaches FCFS, while tiny quanta increase context-switch overhead. Selecting q therefore trades responsiveness against overhead, matching B. Priority management, deadlock policy and memory allocation are separate concerns rather than the defining parameter-selection challenge of ordinary Round Robin. The useful quantum depends on workloads and switching cost.

Why each choice is right or wrong

A - Incorrect: Basic RR uses cyclic ready-queue turns rather than requiring a hierarchy of priority levels.

B - Correct: Quantum length controls responsiveness and switching overhead, creating RR's characteristic tuning tradeoff.

C - Incorrect: Deadlock avoidance concerns resources and dependencies, not the defining RR quantum parameter.

D - Incorrect: Memory allocation is a separate management problem rather than the Round Robin selection rule's main parameter.

Primary note proof

Study-note section OS18

> A large quantum approaches FCFS; tiny quanta increase switch overhead.

Study-note section OS18

> With n continuously ready jobs, no priority bypass and zero overhead, a just-requeued job waits at most (n−1)q for its next turn. This bounds one interval, not total lifetime waiting when arrivals or blocking are unrestricted. Include context-switch costs if given.

Question sources

Operating Systems review questions · Question 13

Question 197 | NS0697

What is a Semaphore?

A. A scheduling algorithm

B. A memory management technique

C. A synchronization variable that controls access to shared resources in a multi-process environment

D. A type of file allocation method

Hint

Track available permits and blocked operations using the stated semaphore model.

Answer C: A synchronization variable with atomic wait and signal operations.

OS11 defines a semaphore through atomic wait/P/down and signal/V/up operations. Tokens control whether participants can proceed with shared resource work, with binary or counting forms. C identifies that synchronization role. Scheduling, memory-management and file-allocation policies solve different resource-selection/layout problems and do not define semaphore semantics.

Why each choice is right or wrong

A - Incorrect: A semaphore synchronizes participants; it is not itself a CPU-selection scheduling algorithm.

B - Incorrect: It controls coordination through permits, rather than choosing memory locations or translation layouts.

C - Correct: Atomic wait/signal on a synchronization variable controls permits for concurrent shared-resource operations.

D - Incorrect: File allocation chooses storage-block layout; that is separate from semaphore synchronization semantics.

Primary note proof

Study-note section OS11

> A semaphore provides atomic wait/P/down and signal/V/up operations. Under the nonnegative-token model, wait consumes one available token or blocks when none is available; signal supplies a token and may wake a waiter. Implementations using negative internal counts encode waiters differently, so state the model. A binary semaphore represents one permit; a counting semaphore tracks several. A mutex normally has ownership rules that a general semaphore lacks.

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt.

Study-note section OS28

> A contiguous allocation gives a process one continuous region.

Study-note section OS38

> Contiguous allocation uses consecutive blocks; logical block i is start+i and supports direct access, but growth and external fragmentation are problems.

Question sources

Operating Systems review questions · Question 14

Question 198 | NS0698

What is the difference between a Binary Semaphore and a Counting Semaphore?

A. Binary semaphores are faster

B. Binary semaphores have only two states (0 or 1), while counting semaphores track available resources up to a maximum count

C. Counting semaphores are used only for deadlock prevention

D. There is no difference

Hint

Track available permits and blocked operations using the stated semaphore model.

Answer B: Binary tracks one permit; counting tracks several.

OS11 defines a binary semaphore as one permit and a counting semaphore as several, under the nonnegative-token model. Thus binary availability is 0/1 while a resource pool can have a larger count. B states this logical distinction. Internal negative-waiter encodings are implementation details, not a reason to claim speed guarantees, deadlock-only usage or no difference.

Why each choice is right or wrong

A - Incorrect: The note gives no universal relative-speed guarantee; permit capacity is the defining distinction.

B - Correct: In the nonnegative-token model, binary availability is zero/one while counting semaphores track several available permits.

C - Incorrect: Counting semaphores coordinate resource pools and are not limited to preventing deadlock.

D - Incorrect: The allowed permit count is a substantive difference explicitly stated here.

Primary note proof

Study-note section OS11

> A semaphore provides atomic wait/P/down and signal/V/up operations. Under the nonnegative-token model, wait consumes one available token or blocks when none is available; signal supplies a token and may wake a waiter. Implementations using negative internal counts encode waiters differently, so state the model. A binary semaphore represents one permit; a counting semaphore tracks several. A mutex normally has ownership rules that a general semaphore lacks.

Question sources

Operating Systems review questions · Question 15

Question 199 | NS0699

What must hold simultaneously for a deadlock to occur?

A. Only Mutual Exclusion and Circular Wait

B. Any two of the four conditions

C. All four necessary conditions: Mutual Exclusion, Hold-and-Wait, No Preemption, and Circular Wait

D. Only Circular Wait

Hint

Trace the resource dependencies and distinguish necessary conditions from evidence of an actual deadlock.

Answer C: All four standard necessary conditions.

OS22 says all four conditions are necessary in the standard reusable-resource model. C names that full set. Any two or a cycle alone does not state the complete conjunction. This is a necessary-condition answer, not an assertion that merely allowing all four policies proves a deadlock already exists in a particular allocation state.

Why each choice is right or wrong

A - Incorrect: It omits hold-and-wait and no-preemption, both necessary in the stated standard resource model.

B - Incorrect: An arbitrary pair does not supply all four necessary restrictions for the standard deadlock model.

C - Correct: This option lists mutual exclusion, hold-and-wait, no-preemption and circular wait together.

D - Incorrect: A cycle alone omits other necessary restrictions; it is not the complete standard condition set.

Primary note proof

Study-note section OS22

> Mutual exclusion allows at most one holder of a nonshareable resource. Hold-and-wait means retaining a resource while requesting another. No resource preemption means release must be voluntary. Circular wait forms a closed chain of such waits.

Study-note section OS22

> All four are necessary in the standard reusable-resource model. Their availability as general system policies does not prove a deadlock exists at this instant; inspect actual allocations/requests.

Question sources

Operating Systems review questions · Question 16

Question 200 | NS0700

In a Resource Allocation Graph (RAG), what does a cycle with single-instance resources indicate?

A. Potential deadlock only

B. No deadlock possible

C. Definite deadlock

D. High CPU utilization

Hint

Follow request and assignment arrows and check how many instances each resource has.

Answer C: Definite deadlock under single-instance resource assumptions.

OS23 says a cycle implies deadlock when every resource type has one instance. The stem supplies that decisive assumption, so C follows. With multiple instances a cycle would indicate only a possibility, but importing that different model would ignore the literal question. A graph cycle also says nothing by itself about high CPU utilization.

Why each choice is right or wrong

A - Incorrect: Potential-only applies to the more general multiple-instance caveat, not the supplied single-instance case.

B - Incorrect: A cycle in the stated single-instance model establishes deadlock rather than excluding it.

C - Correct: The note explicitly gives cycle implies deadlock for a graph with one instance per resource type.

D - Incorrect: Resource dependency cycles do not measure CPU busy time or establish high processor utilization.

Primary note proof

Study-note section OS23

> In a resource-allocation graph, P→R is a request and an instance R→P is an assignment. Follow these directions to form waits. No cycle implies no deadlock; with one instance of every resource type, a cycle implies deadlock. With multiple instances a cycle alone is insufficient: another instance may permit completion. A wait-for graph replaces single-instance resource nodes with P→Q when P waits for Q.

Question sources

Operating Systems review questions · Question 17

Question 201 | NS0701

Which deadlock handling method allows deadlock to occur but then recovers from it?

A. Deadlock Prevention

B. Deadlock Ignorance

C. Deadlock Recovery

D. Deadlock Avoidance

Hint

Separate handling before allocation from actions taken after detection, and consider the cost of checking.

Answer C: Deadlock recovery.

OS26 describes recovery through terminating selected processes or safely reclaiming resources and rolling work back after deadlock is detected. That matches allowing occurrence and then resolving it. Prevention removes a necessary condition, avoidance preserves safety before granting, and ignoring accepts risk without committing to recovery. The complete handling strategy is detection plus recovery; C is the offered recovery component.

Why each choice is right or wrong

A - Incorrect: Prevention constrains requests so the targeted deadlock class cannot arise rather than accepting occurrence first.

B - Incorrect: Ignoring rare deadlock does not define active detection and subsequent recovery.

C - Correct: Recovery acts on an actual detected deadlock using termination, reclamation or rollback as appropriate.

D - Incorrect: Avoidance checks safe states before grants rather than deliberately permitting deadlock and then repairing it.

Primary note proof

Study-note section OS26

> Detection asks whether the current state is deadlocked. Recovery can terminate selected processes or reclaim safely preemptable resources and roll affected work back to a checkpoint. Termination and rollback are both possible recovery tools; choose according to the resource and process conditions.

Study-note section OS24

> Prevention constrains how requests are made.

Study-note section OS25

> Avoidance allows requests only if the resulting state remains safe.

Study-note section OS26

Ignoring a sufficiently rare problem is sometimes called the ostrich policy, but real systems still prevent or detect selected deadlocks.

Question sources

Operating Systems review questions · Question 18

Question 202 | NS0702

What is Address Binding at 'Load Time'?

A. Addresses are fixed by the compiler before execution

B. Addresses are determined while the program executes

C. Final memory addresses are assigned when the program is loaded into memory (since addresses are unknown at compile time)

D. Addresses are stored on the hard disk

Hint

Track when an address or library reference is resolved and the direction of translation.

Answer C: Relocate addresses when the program is loaded.

OS27 defines load-time binding as relocation when placed into memory. That matches C's final-address assignment at load when placement was unknown at compile time. Fixing absolute addresses during compilation is compile-time binding; translating while running is execution-time binding. Merely storing address bytes on a disk describes neither the placement decision nor its timing.

Why each choice is right or wrong

A - Incorrect: Fixing absolute addresses in the compiler is compile-time binding, not relocation on loading.

B - Incorrect: Translation during execution is execution-time binding rather than assignment during loading.

C - Correct: The loader resolves placement-dependent addresses when placing the program into memory.

D - Incorrect: Storing address information on disk does not define when placement-dependent address binding occurs.

Primary note proof

Study-note section OS27

> Compile-time binding uses absolute addresses if placement is known. Load-time binding relocates when placed into memory. Execution-time binding translates while running and permits movement with hardware support.

Question sources

Operating Systems review questions · Question 19

Question 203 | NS0703

Under basic page-frame allocation and variable-segment allocation, which statement correctly contrasts allocation-unit size and external fragmentation?

A. Paging uses variable-sized blocks; segmentation uses fixed-sized blocks

B. Paging divides memory into fixed-sized pages eliminating external fragmentation; segmentation uses variable-sized logical segments but can cause external fragmentation

C. Segmentation is used only in virtual memory

D. Paging requires no hardware support

Hint

Check the logical region and offset bound before computing its physical location.

Answer B: Fixed-size paging avoids external page-frame fragmentation; variable segments can fragment externally.

Paging maps fixed-size pages into equal-size frames, avoiding external fragmentation of frame allocation, although internal waste can remain. Segmentation uses variable-length logical regions and can leave external holes. This is the unit-size and fragmentation contrast in B. A reverses the sizes. C does not supply that contrast, and D substitutes a hardware-support claim for it; page translation still requires mappings and checks.

Why each choice is right or wrong

A - Incorrect: The sizes are reversed: pages are fixed-size units and segments are variable-length regions.

B - Correct: This gives the required fixed-page versus variable-segment size and external-fragmentation contrast.

C - Incorrect: An exclusive-use statement does not compare allocation-unit sizes or external fragmentation as requested.

D - Incorrect: This does not give the requested size/fragmentation contrast; paging also uses translation mappings and checks.

Primary note proof

Study-note section OS31

> Under the basic models, paging uses fixed units/page tables and can waste space internally; segmentation uses variable logical units/segment tables and can fragment externally. Segments reflect program structure whereas ordinary pages are largely transparent to applications.

Study-note section OS30

> Pages and physical frames have equal fixed size P. Divide virtual address v into page=floor(v/P) and offset=v mod P. The page table maps a page to a frame and protection/status flags. Physical address=frame×P+offset.

Question sources

Operating Systems review questions · Question 20

Question 204 | NS0704

What causes External Fragmentation in memory management?

A. Allocated memory blocks being larger than needed

B. Free memory being scattered in small non-contiguous pieces making it impossible to satisfy large allocation requests

C. Processes using too much CPU time

D. Too many open files in the directory

Hint

Locate unused space relative to an allocated unit and distinguish total free space from contiguous free space.

Answer B: Free capacity is split across holes too small for a required contiguous allocation.

OS29 defines external fragmentation as sufficient total free space split into holes unable to satisfy a contiguous allocation. B matches that definition, with large requests understood to require one continuous region. A instead describes internal fragmentation. CPU time and open-file counts may create other pressures but do not define the spatial fragmentation of free memory.

Why each choice is right or wrong

A - Incorrect: Unused space within an oversized allocated block is internal fragmentation, not external free-hole fragmentation.

B - Correct: Noncontiguous small holes can have enough total capacity yet fail to satisfy one contiguous request.

C - Incorrect: CPU execution time does not define the spatial layout of free memory holes.

D - Incorrect: Open-file count is a resource-use limit rather than the definition of external memory fragmentation.

Primary note proof

Study-note section OS29

> Internal fragmentation is unused space inside an allocated unit. External fragmentation is sufficient total free space split into holes that cannot satisfy a required contiguous allocation.

Question sources

Operating Systems review questions · Question 21

Question 205 | NS0705

What is the purpose of the Page Table in Paging?

A. To allocate CPU time to processes

B. To map logical page numbers to physical frame numbers in memory

C. To store file access permissions

D. To manage disk I/O operations

Hint

Split the address into page and offset, then apply the mapping and access checks.

Answer B: Map virtual/logical pages to physical frames.

OS30 says the page table maps each virtual page to a frame with protection/status flags. B directly states the primary translation role. CPU time allocation belongs to scheduling; file permissions belong to filesystem metadata, despite page tables also having memory-protection bits. Disk I/O may service nonresident pages but does not define the page table's mapping purpose.

Why each choice is right or wrong

A - Incorrect: CPU time allocation is scheduling, not the page-to-frame translation recorded by a page table.

B - Correct: The table identifies the physical frame corresponding to each virtual page and records relevant flags.

C - Incorrect: Memory access-protection flags differ from filesystem file-access permissions; file permissions are not the page table's purpose.

D - Incorrect: Page faults may cause I/O, but general disk-operation management is not the table's translation role.

Primary note proof

Study-note section OS30

> Pages and physical frames have equal fixed size P. Divide virtual address v into page=floor(v/P) and offset=v mod P. The page table maps a page to a frame and protection/status flags.

Study-note section OS36

> A file has content and metadata such as name, size, owner, permissions and timestamps. open obtains an open-file reference, read/write transfer bytes, seek changes an offset, close releases a reference, truncate changes size and unlink removes a name.

Question sources

Operating Systems review questions · Question 22

Question 206 | NS0706

In Segmentation, how is a physical address calculated?

A. Page number x frame size

B. Base address of segment + offset

C. Logical address - limit register

D. Frame number + page offset

Hint

Check the logical region and offset bound before computing its physical location.

Answer B: Physical address = segment base + valid offset.

OS31 gives segment-table base, limit and permissions, then validates (segment,offset) and adds base+offset. B matches, conditional on the offset passing the bound/protection checks. A and D mix in paging components and omit required scaling or displacement, while subtracting a limit is not the segment translation formula. An invalid offset traps rather than spilling into another segment.

Why each choice is right or wrong

A - Incorrect: Page number times frame size is not the segment base-plus-offset translation and omits required mapping.

B - Correct: After segment and bound checks, add the selected segment's physical base to the offset.

C - Incorrect: The limit is a bound for validation, not a number subtracted to form the physical address.

D - Incorrect: Frame terminology belongs to paging, and adding an unscaled frame number is not the segmentation formula.

Primary note proof

Study-note section OS31

> Segmentation uses logical variable-length units such as code, stack and arrays. A segment-table entry provides base, limit and permissions. For (segment,offset), validate the segment and check the bound, then add base+offset. An out-of-range offset traps; it must not spill into the next segment.

Study-note section OS30

> Pages and physical frames have equal fixed size P. Divide virtual address v into page=floor(v/P) and offset=v mod P. The page table maps a page to a frame and protection/status flags. Physical address=frame×P+offset.

Question sources

Operating Systems review questions · Question 23

Question 207 | NS0707

What is Virtual Memory?

A. Extra RAM chips added to a computer

B. A technique using hard disk space as an extension of physical memory, allowing programs larger than physical memory to execute

C. The memory inside the CPU cache

D. Memory used only by the operating system

Hint

Separate a permitted nonresident access from an illegal access and trace the required handling steps.

Answer B: Translated protected address spaces can use backing storage for nonresident portions.

OS32 describes virtual memory as protected translated address spaces with nonresident program portions backed by storage. B gives the disk-backed extension aspect and can allow programs exceeding resident physical memory to execute. It is not a complete definition of virtual memory, nor does it eliminate finite capacity constraints. The other alternatives confuse RAM installation, cache or an OS-only restriction.

Why each choice is right or wrong

A - Incorrect: Installing RAM chips adds physical capacity; virtual-memory translation is a different mechanism.

B - Correct: Backing storage holds nonresident portions, permitting a program's address space to exceed its currently resident physical memory.

C - Incorrect: CPU cache retains copied memory blocks and is not the virtual-address/backing-store mechanism.

D - Incorrect: Processes receive virtual address spaces; virtual memory is not restricted to storage used only by the OS.

Primary note proof

Study-note section OS32

> Virtual memory provides translated, protected address spaces and allows nonresident portions of a program to be backed by storage. It does not add physical RAM or remove all capacity constraints.

Study-note section OS35

> Caching retains copies to accelerate reuse.

Question sources

Operating Systems review questions · Question 24

Question 208 | NS0708

In demand paging, assume the accessed virtual page is valid, permitted and backed by disk, but its contents are not resident and must be read from disk. What event is the page fault in this scenario?

A. A hardware memory error

B. When the CPU tries to access a page not currently in physical memory, requiring the OS to load it from disk

C. An error in the page table

D. When a process exceeds its allocated memory

Hint

Separate a permitted nonresident access from an illegal access and trace the required handling steps.

Answer B: When the CPU tries to access a page not currently in physical memory, requiring the OS to load it from disk

A valid nonresident access traps so the OS can obtain the page contents, update the mapping and restart the access. The specified page is disk-backed and requires a read, making B accurate for this scenario. Page faults in general need not read disk; zero-fill and shared mappings are other cases. Nothing here requires hardware failure, table corruption or exceeding the permitted address range.

Why each choice is right or wrong

A - Incorrect: The trap does not imply damaged memory hardware.

B - Correct: The valid disk-backed page is nonresident, so this access requires the stated page-in.

C - Incorrect: A correct table may intentionally mark a valid page nonresident.

D - Incorrect: The page is explicitly within the valid permitted mapping, so no allocation violation is implied.

Primary note proof

Study-note section OS32

> Demand paging brings a page in when accessed. On a nonresident access: trap, validate mapping/permissions, find a frame or choose a victim, save a dirty victim if needed, obtain contents, update mapping and restart.

Study-note section OS32

> Not every fault reads a disk: some allocate zero-filled pages or establish shared mappings. An illegal access is rejected, not automatically loaded.

Question sources

Operating Systems review questions · Question 25

Question 209 | NS0709

What is Demand Paging?

A. Loading the entire program into memory before execution

B. Loading only the pages that are actually needed during program execution, keeping others on disk until required

C. Allocating memory in fixed partitions

D. Swapping entire processes between memory and disk

Hint

Separate a permitted nonresident access from an illegal access and trace the required handling steps.

Answer B: Bring a page into memory when it is needed/accessed.

OS32 defines demand paging as bringing a page in when accessed. B matches loading only the pages demanded during execution, with disk backing for the described stored portions. Not all pages must literally originate on disk because zero-fill/shared cases exist. Whole-program loading, fixed contiguous partitions and whole-process swapping do not name this per-page demand policy.

Why each choice is right or wrong

A - Incorrect: Loading the entire program before execution is eager loading rather than fetching pages on demand.

B - Correct: Pages are brought in as execution needs them; disk backing is the described common stored-page case.

C - Incorrect: Fixed partitions are a contiguous-allocation arrangement rather than the page-access-triggered loading policy.

D - Incorrect: Moving an entire process is traditional swapping, not the individual-page demand mechanism.

Primary note proof

Study-note section OS32

> Demand paging brings a page in when accessed. On a nonresident access: trap, validate mapping/permissions, find a frame or choose a victim, save a dirty victim if needed, obtain contents, update mapping and restart.

Study-note section OS27

> Swapping traditionally moves a whole process to backing store; paging moves individual pages.

Study-note section OS28

> A contiguous allocation gives a process one continuous region.

Question sources

Operating Systems review questions · Question 26

Question 210 | NS0710

What is the difference between Internal and External Fragmentation?

A. Both are the same type of memory waste

B. Internal fragmentation wastes space inside allocated partitions (allocated > needed); external fragmentation wastes space between allocations

C. External fragmentation occurs in paging systems

D. Internal fragmentation only occurs in virtual memory

Hint

Locate unused space relative to an allocated unit and distinguish total free space from contiguous free space.

Answer B: Internal waste is inside allocations; external holes are between them.

OS29 defines internal fragmentation as unused space inside an allocated unit and external fragmentation as free holes unable to serve a required contiguous request. B states the location distinction. C is not the general definition and needs allocator scope; D adds an unsupported virtual-memory-only restriction. These wastes can arise from allocation geometry rather than the program's CPU behavior.

Why each choice is right or wrong

A - Incorrect: The note explicitly distinguishes internal unused allocation space from external free-hole fragmentation.

B - Correct: Internal space is wasted within an allocation; external waste is free space split between allocations.

C - Incorrect: This does not define the contrast; basic page-frame allocation avoids external fragmentation, though other allocators may differ.

D - Incorrect: Unused space within an allocated unit is not limited by definition to virtual-memory systems.

Primary note proof

Study-note section OS29

> Internal fragmentation is unused space inside an allocated unit. External fragmentation is sufficient total free space split into holes that cannot satisfy a required contiguous allocation.

Study-note section OS29

> Basic fixed-frame paging avoids external fragmentation of page-frame allocation but can waste the final page internally.

Question sources

Operating Systems review questions · Question 27

Question 211 | NS0711

Which memory allocation strategy picks the first hole large enough to accommodate a process?

A. Best-Fit

B. Worst-Fit

C. First-Fit

D. Next-Fit

Hint

Apply the stated hole-selection or base-and-bound rule with its explicit convention.

Answer C: First-fit.

OS28 defines first-fit as selecting the first adequate hole. This is exactly the stem's first hole large enough. Best-fit searches for the smallest adequate hole, worst-fit for the largest, and next-fit resumes from a prior search position. The explicit first-hole condition selects first-fit without any numeric hole trace.

Why each choice is right or wrong

A - Incorrect: Best-fit minimizes adequate hole size rather than stopping at the first adequate hole.

B - Incorrect: Worst-fit selects the largest available hole rather than the first adequate one encountered.

C - Correct: First-fit stops at the first hole that can accommodate the requested allocation.

D - Incorrect: Next-fit continues from the previous search position; the stem gives the ordinary first-fit criterion.

Primary note proof

Study-note section OS28

> First-fit selects the first adequate hole; next-fit continues from the previous search position; best-fit selects the smallest adequate hole; worst-fit selects the largest.

Question sources

Operating Systems review questions · Question 28

Question 212 | NS0712

What is Swapping in OS memory management?

A. Exchanging CPU registers between processes

B. Temporarily moving an entire process from main memory to a backing store (disk) to free memory for other processes

C. Copying files between directories

D. Replacing old CPU instructions with new ones

Hint

Track when an address or library reference is resolved and the direction of translation.

Answer B: Temporarily move a whole process to backing store.

OS27 explicitly distinguishes traditional swapping of a whole process to backing store from paging individual pages and context switching execution state. B describes the whole-process memory-management operation. Saving registers is a context-switch task, file copying is a filesystem operation, and replacing instruction contents does not define swapping's residence change.

Why each choice is right or wrong

A - Incorrect: Saving/restoring CPU register contexts is context switching rather than moving a whole process out of RAM.

B - Correct: Traditional swapping moves a process image to backing storage to free resident memory.

C - Incorrect: Copying files changes directory/storage contents rather than a process image's memory residence.

D - Incorrect: Replacing instruction contents is not the defined transfer of a whole process between RAM and backing store.

Primary note proof

Study-note section OS27

> Swapping traditionally moves a whole process to backing store; paging moves individual pages. These differ from a context switch, which primarily changes execution state.

Study-note section OS05

> A context switch saves one execution context and restores another; useful application work is interrupted by this overhead.

Question sources

Operating Systems review questions · Question 29

Question 213 | NS0713

What is Dynamic Loading?

A. Loading the entire program into memory at startup

B. Loading routines from disk into memory only when they are actually called during execution

C. Dynamically changing the priority of a process

D. Moving pages between RAM and disk automatically

Hint

Track when an address or library reference is resolved and the direction of translation.

Answer B: Load a routine only when needed.

OS27 directly defines dynamic loading as bringing a routine in when needed. B states the routine-on-call form of that rule. Eager startup loading is the opposite, priority changes are scheduling, and automatic page movement is a virtual-memory mechanism at a different granularity. Dynamic loading concerns when code routines become resident, not which process gets CPU priority.

Why each choice is right or wrong

A - Incorrect: Loading all routines at startup is eager loading rather than bringing them in only on demand.

B - Correct: A routine is loaded when execution actually calls or needs it, matching dynamic loading.

C - Incorrect: Changing priority affects scheduling and does not load a routine's code into memory.

D - Incorrect: Paging moves pages by virtual-memory demand, whereas dynamic loading names demand-driven routine loading.

Primary note proof

Study-note section OS27

> Dynamic loading brings a routine in when needed.

Study-note section OS32

> Demand paging brings a page in when accessed.

Study-note section OS17

> Priority scheduling chooses the highest-priority ready job.

Question sources

Operating Systems review questions · Question 30

Question 214 | NS0714

What is the key management-layer distinction between user-level threads and kernel-level threads, without assuming universal speed or reliability advantages?

A. User-level threads are managed by a user-space library; kernel-level threads are known to and managed by the OS

B. Kernel-level threads are always faster

C. User-level threads can access hardware directly

D. There is no practical difference between them

Hint

Compare per-thread execution context with shared process resources and the stated thread mapping.

Answer A: User-level threads are managed by a user-space library; kernel-level threads are known to and managed by the OS

User-level threads are managed by a library, while kernel threads are execution contexts known to the OS. Their mapping affects blocking behavior and use of multiple CPUs. The management layer alone does not guarantee faster creation, faster execution or greater stability; those depend on implementation. It also does not grant user code privileged hardware access. The option keeps the supported management distinction.

Why each choice is right or wrong

A - Correct: This states library management versus OS-known kernel threads.

B - Incorrect: No management layer guarantees that kernel threads are always faster.

C - Incorrect: User-level thread management does not grant direct privileged hardware access.

D - Incorrect: Blocking and CPU-use differences show that the distinction has practical effects.

Primary note proof

Study-note section OS08

> User-level threads are managed by a library; kernel threads are known to the OS. In many-to-one mapping, a blocking kernel call can block all user threads and one kernel execution context cannot use multiple CPUs simultaneously. One-to-one permits another kernel thread to run when one blocks. Many-to-many multiplexes user threads over several kernel threads. Speed and reliability depend on implementation; the management layer alone guarantees neither. Application names such as browser or Excel do not prove a particular version’s threading model.

Study-note section OS-S02

> Being scheduled or managed by the kernel does not give application thread code kernel privilege. Ordinary applications and supporting utilities execute in restricted user mode and request protected services through libraries and system calls. Kernel component code executes in privileged kernel mode and performs permitted low-level hardware operations.

Question sources

Operating Systems review questions · Question 31

Question 215 | NS0715

What is IPC (Inter-Process Communication)?

A. A CPU scheduling algorithm

B. Mechanisms allowing processes to exchange information, synchronize, and coordinate activities

C. A method for allocating memory between processes

D. A file system operation

Hint

Identify how data moves, which participant waits and what event allows the operation to continue.

Answer B: Mechanisms for exchanging information and coordinating processes.

OS09 describes shared-memory communication, send/receive, pipes, queues, sockets and RPC, including synchronization for conflicting updates. B summarizes the exchange and coordination role. IPC is not a CPU selection algorithm or a particular memory-allocation/file operation, even though shared memory and file-based mechanisms may be used to implement it.

Why each choice is right or wrong

A - Incorrect: CPU scheduling selects execution opportunities, rather than defining communication and coordination among processes.

B - Correct: Exchanging information with appropriate synchronization/coordination captures the IPC mechanisms described here.

C - Incorrect: Shared memory can implement IPC, but memory allocation alone does not define interprocess communication.

D - Incorrect: Files can participate in communication, but IPC is not restricted to one filesystem operation.

Primary note proof

Study-note section OS09

> Shared memory lets processes access a common region and requires synchronization for conflicting updates. Message passing exchanges data through send/receive operations. Pipes provide streams; message queues preserve messages; sockets connect endpoints; RPC requests a remote operation. HTTP is an application protocol over a communication mechanism and can carry interprocess communication, so it is not universally “not IPC.”

Study-note section OS09

File locks coordinate access rather than transfer arbitrary payloads, though locking is synchronization within IPC.

Question sources

Operating Systems review questions · Question 32

Question 216 | NS0716

What is the Critical Section problem concerned with?

A. Preventing processes from using too much CPU

B. Ensuring that no two processes are executing their critical sections (code accessing shared resources) simultaneously

C. Managing disk I/O operations

D. Scheduling processes in real-time systems

Hint

Identify conflicting shared accesses and the safety or progress property required.

Answer B: Mutually exclude conflicting critical-region execution.

OS10 defines a critical section as code accessing shared state needing protection and lists mutual exclusion, progress and bounded waiting as requirements. B states the exclusion aspect for critical sections protecting the same conflicting resource. The problem also includes progress/fair-waiting requirements, so exclusion is not its entire specification. CPU quotas, disk management and real-time scheduling are different topics.

Why each choice is right or wrong

A - Incorrect: CPU consumption limits are not the shared-access exclusion problem defining critical sections.

B - Correct: Conflicting critical regions must not execute simultaneously; this is the mutual-exclusion part of the correctness requirements.

C - Incorrect: Disk I/O can involve shared state, but managing it generally is not the critical-section problem definition.

D - Incorrect: Real-time selection addresses timing/scheduling, not the core shared-state exclusion requirement requested.

Primary note proof

Study-note section OS10

A race occurs when results depend on uncoordinated interleaving of conflicting accesses. A critical section is code accessing shared state that needs protection, not the resource itself. A correct exclusion solution requires mutual exclusion, progress and bounded waiting under stated assumptions. Mutual exclusion is a desired safety property; races, starvation and deadlock are possible failures.

Study-note section OS10

Keep critical regions small and ensure every conflicting access uses the same protection. A barrier waits until a defined set of participants reaches a point; a monitor encapsulates protected operations with condition synchronization. Locking can itself introduce deadlock if acquisition order is inconsistent.

Question sources

Operating Systems review questions · Question 33

Question 217 | NS0717

For a general process that may perform I/O, which formula defines turnaround time from event timestamps? Waiting means time in the ready queue; burst means total CPU service. The formula must remain valid for nonzero arrival times and blocked intervals.

A. Burst Time - Waiting Time

B. Completion Time - Arrival Time

C. Start Time + Burst Time

D. Waiting Time + Burst Time

Hint

Construct a timeline and distinguish completion rates from individual elapsed-time measures.

Answer B: Completion Time - Arrival Time

Turnaround is completion time minus arrival time, including CPU execution, ready-queue waiting and blocked intervals. Ready waiting plus CPU service omits blocked time when I/O occurs. Start plus burst may resemble a completion timestamp for uninterrupted work, but it is not elapsed turnaround and does not subtract arrival. Burst minus waiting has no general elapsed-time interpretation. Only B remains valid in the full stated model.

Why each choice is right or wrong

A - Incorrect: Subtracting ready waiting from CPU service is not turnaround.

B - Correct: Completion minus arrival includes the entire elapsed lifetime.

C - Incorrect: Start plus burst is not generally completion with I/O and does not subtract arrival.

D - Incorrect: Ready waiting plus CPU service omits blocked intervals.

Primary note proof

Study-note section OS13

> For arrival A, first start S and completion C, response=S−A and turnaround=C−A. For a single CPU burst B with no I/O, waiting=C−A−B. With I/O, subtract all CPU and blocked time to isolate ready-queue waiting.

Question sources

Operating Systems review questions · Question 34

Question 218 | NS0718

What is the Convoy Effect in FCFS scheduling?

A. High-priority processes always complete first

B. All processes receive equal CPU time

C. Short processes get stuck waiting behind one very long process, causing high average waiting times

D. Multiple processes share the same memory space

Hint

Follow arrival order, include idle periods and distinguish waiting from completion time.

Answer C: Short jobs wait behind a long FCFS burst.

OS15 describes the convoy effect as long CPU-bound work making many short jobs wait under non-preemptive FCFS. C directly states that queueing effect and its high waiting-time consequence. FCFS does not use priority order or grant equal total CPU, and shared address space is unrelated to the arrival-order delay. The effect depends on a long burst preceding shorter ones.

Why each choice is right or wrong

A - Incorrect: Priority order does not define FCFS, which follows arrival order rather than high-priority completion.

B - Incorrect: FCFS serves entire bursts of possibly different lengths rather than enforcing equal CPU shares.

C - Correct: Short jobs are delayed behind the long non-preempted burst, producing the FCFS convoy effect.

D - Incorrect: Shared memory concerns address-space organization rather than the scheduling delay called a convoy.

Primary note proof

Study-note section OS15

> FCFS runs ready bursts in arrival order and ordinarily does not preempt. If the CPU is idle, jump to the next arrival. Long CPU-bound work can make many short jobs wait: the convoy effect.

Study-note section OS18

> RR gives recurring opportunities, not identical total CPU consumption for jobs of different lengths.

Study-note section OS08

> A thread has its own PC, registers and stack. Threads in a process share code, address space and usually open-file resources.

Question sources

Operating Systems review questions · Question 35

Question 219 | NS0719

What is Cooperative (Non-Preemptive) vs Preemptive Scheduling?

A. They are the same thing

B. Non-preemptive: process runs until it voluntarily releases CPU; Preemptive: OS can forcibly take CPU from a running process for another

C. Preemptive scheduling never interrupts processes

D. Non-preemptive scheduling always gives better performance

Hint

Check whether the running burst can be interrupted before completion or blocking.

Answer B: Voluntary release versus possible forced CPU preemption.

OS14 says non-preemptive scheduling retains the CPU until completion, blocking or voluntary yielding, while preemptive scheduling may return running work to ready. B captures the voluntary-versus-forced distinction, with completion/blocking included among natural release events. C contradicts preemption, D claims an unsupported universal performance ranking, and A erases the defining difference.

Why each choice is right or wrong

A - Incorrect: The policies differ in whether a running CPU burst can be forcibly interrupted for another process.

B - Correct: Non-preemptive work releases naturally/voluntarily; preemptive policy may forcibly reassign CPU execution time.

C - Incorrect: Forcibly interrupting running work is exactly what preemptive scheduling permits.

D - Incorrect: No policy universally performs better for all workloads and objectives, so the absolute performance claim fails.

Primary note proof

Study-note section OS14

> Non-preemptive scheduling keeps a running CPU burst until completion, blocking or voluntary yielding. Preemptive scheduling can return running work to ready, usually after a timer quantum or a more urgent arrival. FCFS and non-preemptive SJF are non-preemptive; RR and SRTF are preemptive. Priority can be either, so its name alone cannot decide. Cooperative describes relying on voluntary release.

Study-note section OS21

> There is no universally best policy for response, waiting, fairness and throughput simultaneously.

Question sources

Operating Systems review questions · Question 36

Question 220 | NS0720

Which scheduling criterion measures the number of processes completed per unit of time?

A. CPU Utilization

B. Response Time

C. Throughput

D. Turnaround Time

Hint

Construct a timeline and distinguish completion rates from individual elapsed-time measures.

Answer C: Throughput.

OS13 defines throughput as completed jobs divided by elapsed time. That exactly matches the stem's process-completion rate. CPU utilization is busy-time fraction; response measures delay to first start, and turnaround delay to completion. The units therefore distinguish throughput from the other criteria without requiring a scheduling example.

Why each choice is right or wrong

A - Incorrect: CPU utilization measures busy-time fraction, not how many processes finish per unit time.

B - Incorrect: Response measures first-service delay for a process rather than a completion-count rate.

C - Correct: Throughput is explicitly the number of completed jobs divided by elapsed time.

D - Incorrect: Turnaround measures arrival-to-completion elapsed duration rather than completed jobs per time.

Primary note proof

Study-note section OS13

> For arrival A, first start S and completion C, response=S−A and turnaround=C−A. For a single CPU burst B with no I/O, waiting=C−A−B. With I/O, subtract all CPU and blocked time to isolate ready-queue waiting. Throughput=completed jobs/elapsed time; CPU utilization=busy time/elapsed time. Minimize response/turnaround/waiting under the selected goal and maximize useful throughput/utilization.

Question sources

Operating Systems review questions · Question 37

Question 221 | NS0721

What does the Job Queue contain?

A. Only processes currently running on the CPU

B. All processes currently in the system, including those not yet admitted to memory

C. Only processes waiting for I/O devices

D. Processes that have already terminated

Hint

Decide whether the process is eligible, running or waiting, then identify the responsible queue or scheduling role.

Answer B: All processes in the broad textbook system/job-queue sense.

OS06 explicitly defines the broad textbook job queue as processes in the system and distinguishes a narrower admission queue awaiting memory. B matches that broad usage, including not-yet-admitted jobs. Ready/running or I/O-blocked work are subsets, not the whole category; already-ended processes are not the intended current job population. The answer retains the note's terminology qualification.

Why each choice is right or wrong

A - Incorrect: Currently running work is only a subset, not the broad population represented by the textbook job queue.

B - Correct: The broad textbook usage includes system processes/jobs, including those awaiting memory admission.

C - Incorrect: I/O waiting belongs to device queues and does not exhaust the broad job-queue population.

D - Incorrect: Already-terminated work is not the current job population intended here's broad system queue.

Primary note proof

Study-note section OS06

> The ready queue contains eligible work; device queues hold blocked requests. The broad textbook job queue contains processes in the system, while an admission queue more narrowly holds jobs awaiting memory/admission. Long-term scheduling admits jobs; medium-term scheduling suspends/swaps them; short-term scheduling selects ready work.

Study-note section OS06

> New processes are created/admitted; ready means eligible for CPU but not currently running. Running uses a CPU. Waiting/blocked means unable to proceed until an event; terminated means execution has ended. Running→ready follows time-slice expiry or another preemption. Running→waiting follows blocking I/O. I/O completion normally gives waiting→ready, not a guaranteed immediate running state.

Question sources

Operating Systems review questions · Question 38

Question 222 | NS0722

In an open seekable file’s state, what does its current file-offset field (the file-position pointer, rather than the descriptor handle) track?

A. The physical location of the file on disk

B. The current position within an open file for the next read or write operation

C. The owner's access permissions

D. The file's creation timestamp

Hint

Distinguish a file operation, a resource handle, an offset and file metadata.

Answer B: The current position within an open file for the next read or write operation

The current offset records the logical position used for the next read or write on a seekable open file. A descriptor is instead a process-local integer handle referring to open-file state. Physical disk-block locations, permissions and creation timestamps describe other mapping or metadata information. By asking specifically about the position field, the revised stem makes B precise without conflating a descriptor with an offset.

Why each choice is right or wrong

A - Incorrect: Physical disk location is separate from the current logical I/O offset.

B - Correct: The file-offset field records the position for the next read or write.

C - Incorrect: Owner permissions are access-control metadata rather than the current position.

D - Incorrect: Creation time is metadata rather than the current I/O offset.

Primary note proof

Study-note section OS36

> A file has content and metadata such as name, size, owner, permissions and timestamps. open obtains an open-file reference, read/write transfer bytes, seek changes an offset, close releases a reference, truncate changes size and unlink removes a name. A file descriptor is a process-local integer handle; the associated open-file state contains the current offset and access mode. Descriptor and offset are not synonyms.

Study-note section OS36

> Read advances the offset on seekable files and may return fewer bytes than requested; zero normally indicates end-of-file.

Question sources

Operating Systems review questions · Question 39

Question 223 | NS0723

What is the purpose of an Access Control List (ACL) in file protection?

A. To track disk space usage

B. To specify exactly which users/groups have which types of access (read, write, execute) to each file

C. To manage disk fragmentation

D. To schedule file I/O operations

Hint

Distinguish identity checks, permission checks and the particular security property at issue.

Answer B: Specify principals and permitted access rights.

OS39 defines an ACL as listing principals and rights. B states that users/groups can receive different read/write/execute permissions for a file. Authentication verifies identity first; authorization uses these permitted operations. Space accounting, fragmentation and I/O ordering do not define an access-control list. The answer does not assume universal default rights or that every filesystem uses exactly the same permission model.

Why each choice is right or wrong

A - Incorrect: Disk-space usage is storage accounting, not the list of principals and allowed operations.

B - Correct: An ACL associates users/groups or other principals with specified access rights on the resource.

C - Incorrect: Fragmentation concerns block placement and free-space layout rather than authorization entries.

D - Incorrect: Scheduling I/O requests controls service order, not which principal is authorized to access a file.

Primary note proof

Study-note section OS39

> Authentication verifies identity; authorization checks permitted operations. An ACL lists principals and rights; owner/group/other bits are a simpler policy model, not universal defaults granting all owners full access or everyone read. Read, write, execute, append and delete are different rights, with filesystem-specific rules.

Study-note section OS29

> Internal fragmentation is unused space inside an allocated unit. External fragmentation is sufficient total free space split into holes that cannot satisfy a required contiguous allocation.

Study-note section OS06

> The ready queue contains eligible work; device queues hold blocked requests.

Question sources

Operating Systems review questions · Question 40

Question 224 | NS0724

What is Contiguous Allocation of disk space?

A. Files are stored in scattered blocks linked by pointers

B. Each file occupies a continuous set of consecutive disk blocks, enabling fast sequential and direct access

C. Files are indexed using a separate index block

D. Files are allocated in fixed-size frames

Hint

Trace how logical file blocks reach physical blocks and account for pointer metadata.

Answer B: A file occupies consecutive disk blocks.

OS38 defines contiguous allocation through consecutive blocks, with logical block i at start+i. This permits sequential and direct access without following a chain. B matches. Scattered pointers describe linked allocation, a separate index describes indexed allocation, and fixed physical memory frames belong to paging rather than file-contiguity organization. Growth and external fragmentation remain drawbacks.

Why each choice is right or wrong

A - Incorrect: Scattered blocks linked by pointers describe linked allocation rather than one consecutive run.

B - Correct: A consecutive block run supports start-plus-index lookup and sequential/direct file access.

C - Incorrect: A separate block-pointer index describes indexed allocation, not contiguous placement itself.

D - Incorrect: Physical memory frames are paging units rather than the disk-block contiguity rule for a file.

Primary note proof

Study-note section OS38

> Contiguous allocation uses consecutive blocks; logical block i is start+i and supports direct access, but growth and external fragmentation are problems. Extents allow several contiguous runs. Linked allocation chains blocks and supports growth without reserving a complete contiguous file; locating block i requires traversing preceding links. FAT stores chains in a table; caching that table does not change the chain into a direct arithmetic mapping.

Study-note section OS38

> Indexed allocation uses a per-file array/tree of block pointers and supports direct lookup with metadata overhead.

Study-note section OS30

> Pages and physical frames have equal fixed size P.

Question sources

Operating Systems review questions · Question 41

Question 225 | NS0725

What problem does Linked Allocation solve compared to Contiguous Allocation?

A. It speeds up random access

B. It eliminates external fragmentation and pre-known file size requirements since blocks can be scattered and linked anywhere

C. It reduces internal fragmentation

D. It requires less memory for storing file metadata

Hint

Trace how logical file blocks reach physical blocks and account for pointer metadata.

Answer B: Linked blocks permit growth without reserving a full contiguous run.

OS38 says linked allocation supports growth without reserving a complete contiguous file. Since successive blocks can occupy separate free locations, it removes the external-contiguity problem and the need to reserve the final length beforehand. B captures that advantage. Random access still requires traversing earlier links, internal last-block waste may remain, and no universal smaller-metadata claim follows.

Why each choice is right or wrong

A - Incorrect: Locating a later linked block requires traversal, so faster random access is not its general advantage.

B - Correct: Separate linked blocks allow incremental growth without a single reserved contiguous region or preknown final length.

C - Incorrect: A partially used final block can still waste space internally; linking does not specifically remove that waste.

D - Incorrect: Pointers add metadata, and the note gives no universal less-memory guarantee over contiguous allocation.

Primary note proof

Study-note section OS38

> Contiguous allocation uses consecutive blocks; logical block i is start+i and supports direct access, but growth and external fragmentation are problems. Extents allow several contiguous runs.

Linked allocation chains blocks and supports growth without reserving a complete contiguous file; locating block i requires traversing preceding links. FAT stores chains in a table; caching that table does not change the chain into a direct arithmetic mapping.

Study-note section OS29

Internal fragmentation is unused space inside an allocated unit. External fragmentation is sufficient total free space split into holes that cannot satisfy a required contiguous allocation.

Question sources

Operating Systems review questions · Question 42

Question 226 | NS0726

In Indexed Allocation, what is the Index Block?

A. A pointer stored in each data block to the next block

B. A dedicated block containing all the disk-block addresses (pointers to data blocks) for a specific file

C. A table stored at the beginning of the disk for all files

D. A block that stores file metadata like size and timestamps

Hint

Trace how logical file blocks reach physical blocks and account for pointer metadata.

Answer B: A per-file block of data-block pointers in the simple indexed model.

OS38 defines indexed allocation through a per-file array/tree of block pointers and gives an index-block capacity example. B describes the simple single-index-block form. Large files may need multiple index levels, so all addresses in one block is model-dependent rather than universal. A is linked allocation, C a global-table description, and D ordinary metadata without the defining data-block pointers.

Why each choice is right or wrong

A - Incorrect: A next pointer in each data block describes linked allocation rather than a separate per-file index.

B - Correct: In the simple indexed model, a dedicated per-file index contains pointers to the file's data blocks.

C - Incorrect: A global table for all files is not the per-file index block described by indexed allocation.

D - Incorrect: Size/timestamps alone do not provide the data-block address index that defines this allocation structure.

Primary note proof

Study-note section OS38

Indexed allocation uses a per-file array/tree of block pointers and supports direct lookup with metadata overhead. A Unix inode can combine direct pointers with indirect levels. Check pointer size, block size, index capacity and whether metadata reads are cached.

Study-note section OS38

A 1024-byte index block with 4-byte pointers holds 256 block addresses. With 1024-byte data blocks, one such index level addresses 256 KiB of file data, excluding metadata and any reserved entries.

Study-note section OS38

FAT stores chains in a table; caching that table does not change the chain into a direct arithmetic mapping.

Question sources

Operating Systems review questions · Question 43

Question 227 | NS0727

What is the difference between Sequential Access and Direct Access for files?

A. They are identical access methods

B. Sequential access reads/writes records in order; direct access allows jumping to any block/record position by specifying its block number

C. Direct access is only possible for text files

D. Sequential access requires more disk space

Hint

Trace how logical file blocks reach physical blocks and account for pointer metadata.

Answer B: Sequential access proceeds in order; direct access selects a position.

OS38 contrasts linked traversal of preceding blocks with contiguous start+i and indexed direct lookup. These illustrate the access distinction in B: sequential progression versus selecting a numbered position without reading every preceding record. Nothing restricts direct access to text or requires sequential files to occupy more disk space merely because of their access method.

Why each choice is right or wrong

A - Incorrect: Order-dependent traversal and direct selection are different access behaviors, not identical methods.

B - Correct: Sequential access advances through records; direct access identifies a chosen block/position for access.

C - Incorrect: The direct block-lookup rule does not impose a text-file-only restriction.

D - Incorrect: Access order alone does not determine a greater storage-space requirement for sequential files.

Primary note proof

Study-note section OS38

> Contiguous allocation uses consecutive blocks; logical block i is start+i and supports direct access, but growth and external fragmentation are problems. Extents allow several contiguous runs. Linked allocation chains blocks and supports growth without reserving a complete contiguous file; locating block i requires traversing preceding links. FAT stores chains in a table; caching that table does not change the chain into a direct arithmetic mapping.

Study-note section OS38

> Indexed allocation uses a per-file array/tree of block pointers and supports direct lookup with metadata overhead.

Question sources

Operating Systems review questions · Question 44

Question 228 | NS0728

What is a Tree-Structured Directory?

A. A directory where all files are in one flat list

B. A hierarchical directory organization with a root directory from which subdirectories and files branch out, each file having a unique path

C. A directory shared among multiple users

D. A temporary directory deleted on reboot

Hint

Follow namespace structure and distinguish names, underlying files and mounted filesystems.

Answer B: A rooted hierarchy of nested directories and files.

OS37 says a tree allows nested directories and absolute/relative paths. In the pure tree model, each entry has a path from the root, matching B. Flat lists instead describe single-level organization; sharing among users and deletion at reboot do not define tree structure. Links can add graph relationships, but that is an extension beyond the pure tree premise.

Why each choice is right or wrong

A - Incorrect: One flat file list is single-level organization rather than nested branches from a root.

B - Correct: A root with nested subdirectories and paths is the tree hierarchy described here.

C - Incorrect: Multiple users sharing a directory does not by itself define its structural hierarchy.

D - Incorrect: Deletion on reboot is a lifetime policy rather than the rooted directory-tree organization.

Primary note proof

Study-note section OS37

> A single-level directory requires unique names in one namespace. Two-level directories separate users. A tree allows nested directories and absolute/relative paths; links can create shared graphs. Modern general-purpose systems commonly present hierarchical namespaces, but a feature name alone does not prove that every system excludes simpler arrangements.

Question sources

Operating Systems review questions · Question 45

Question 229 | NS0729

What is the difference between Hard Links and Symbolic Links in file systems?

A. Hard links can span different file systems; symbolic links cannot

B. Hard links create multiple directory entries pointing to the same file's data blocks; symbolic links are special files containing the path to another file

C. Symbolic links cannot be deleted

D. Hard links are only supported in Windows

Hint

Follow namespace structure and distinguish names, underlying files and mounted filesystems.

Answer B: Hard links name one file object; symbolic links store a pathname.

A hard link supplies another name for the same underlying file object; a symbolic link stores a pathname. Symbolic links can cross filesystems and may dangle, whereas ordinary hard links stay within a filesystem. Linux's link operation supplies a direct counterexample to the Windows-only claim. unlink can remove a symbolic-link name, so the link is not undeletable. B states the correct distinction.

Why each choice is right or wrong

A - Incorrect: The boundary is reversed: symbolic links may cross filesystems, while ordinary hard links remain within one.

B - Correct: Both hard-link names identify the same underlying file; a symbolic link contains a pathname to resolve.

C - Incorrect: A symbolic-link name can be removed with unlink; removing it does not erase the target's contents.

D - Incorrect: Linux provides hard links through link(), so they are not supported only in Windows.

Primary note proof

Study-note section OS37

> A hard link is another name for the same underlying file object, usually within one filesystem; a symbolic link stores a pathname and can point across filesystems or become dangling. Removing one hard link does not erase content while other names/open references retain it.

Study-note section OS-S06

> Linux provides link() to create another hard-link name for an existing file. The old and new names identify the same file, so hard links are not a Windows-only facility. unlink() removes a filesystem name; when that name is a symbolic link, it removes the link itself. Removing a name and erasing the target's contents are different operations.

Question sources

Operating Systems review questions · Question 46

Question 230 | NS0730

What is File System Mounting?

A. Creating a new file system

B. The process of attaching a file system to a specific location in the existing directory tree, making it accessible

C. Deleting old files to free disk space

D. Formatting a disk partition

Hint

Follow namespace structure and distinguish names, underlying files and mounted filesystems.

Answer B: Attach a filesystem at a mount point.

OS37 explicitly defines mounting as attaching a filesystem at a mount point and distinguishes it from formatting. B states that integration into the existing namespace. Creation/formatting establish a filesystem's on-device structure; deleting files frees content space. None of those is the namespace attachment operation requested by mounting.

Why each choice is right or wrong

A - Incorrect: Creating a new filesystem establishes storage structure rather than attaching it to an existing namespace.

B - Correct: Mounting makes a filesystem accessible at a chosen point in the directory hierarchy.

C - Incorrect: Deleting files changes contents/free space rather than attaching a filesystem to the namespace.

D - Incorrect: Formatting is explicitly distinguished from mounting in the supplied note.

Primary note proof

Study-note section OS37

> NFS provides remote filesystem access; FTP transfers files rather than itself mounting a local filesystem.

Study-note section OS35

> Disk initialization/filesystem creation is OS/tool work, firmware begins boot, and drive firmware may remap bad sectors.

Question sources

Operating Systems review questions · Question 47

Question 231 | NS0731

In the context of deadlock recovery, what is 'Rollback'?

A. Restarting the entire operating system

B. Returning a preempted process to a safe previous state after resources are taken back from it during deadlock resolution

C. Deleting all processes involved in a deadlock

D. Rebooting the computer to free all resources

Hint

Separate handling before allocation from actions taken after detection, and consider the cost of checking.

Answer B: Restore an affected process to a prior checkpoint after reclaiming resources.

OS26 describes reclaiming safely preemptable resources and rolling affected work back to a checkpoint. B states that restoration to a safe prior state after resources are withdrawn. Restarting the entire OS/computer and killing all involved processes are different coarse recovery actions, not the meaning of rollback. Safe checkpoints/restart semantics are required to make withdrawal and restoration coherent.

Why each choice is right or wrong

A - Incorrect: Restarting the whole OS is not the checkpoint restoration of affected work that defines rollback.

B - Correct: Rollback restores an affected process to a previous recoverable state after safe resource reclamation.

C - Incorrect: Terminating processes is a separate recovery tool rather than restoring their earlier execution state.

D - Incorrect: Rebooting frees broad machine state but is not the process-checkpoint restoration definition of rollback.

Primary note proof

Study-note section OS26

> Detection asks whether the current state is deadlocked. Recovery can terminate selected processes or reclaim safely preemptable resources and roll affected work back to a checkpoint. Termination and rollback are both possible recovery tools; choose according to the resource and process conditions.

Question sources

Operating Systems review questions · Question 48

Question 232 | NS0732

What is the purpose of relocation/base and limit registers in the standard contiguous single-process-region memory model? The limit denotes the permitted region length.

A. They speed up CPU scheduling decisions

B. The relocation register stores the process's physical base and the limit its permitted length; hardware checks the logical offset before adding the base

C. They track file system locations on disk

D. They manage virtual memory page tables

Hint

Apply the stated hole-selection or base-and-bound rule with its explicit convention.

Answer B: The relocation register stores the process's physical base and the limit its permitted length; hardware checks the logical offset before adding the base

In the standard base/limit model, the base identifies the process's physical starting address and the limit gives its permitted length. Hardware checks that the logical offset is nonnegative and smaller than the limit, then computes base plus offset. This protects accesses to the assigned region. The registers do not describe the OS base and size, schedule CPU work, track disk files or manage page tables.

Why each choice is right or wrong

A - Incorrect: These registers check memory accesses rather than schedule CPU work.

B - Correct: The option states the process base, length bound and checked translation.

C - Incorrect: Disk-block locations are not the purpose of RAM base/limit registers.

D - Incorrect: Simple contiguous base/limit translation does not manage page-table mappings.

Primary note proof

Study-note section OS28

> A contiguous allocation gives a process one continuous region. In the standard base/limit model, base is that process's physical starting address and limit is its permitted length. Check 0≤logical_offset<limit, then physical=base+offset. The relocation register does not normally hold the OS base.

Study-note section OS30

> Pages and physical frames have equal fixed size P. Divide virtual address v into page=floor(v/P) and offset=v mod P. The page table maps a page to a frame and protection/status flags.

Question sources

Operating Systems review questions · Question 49

Question 233 | NS0733

In Dynamic Linking, what is a key advantage over Static Linking?

A. Programs run faster with dynamic linking

B. Multiple programs share one copy of library code in memory rather than each program embedding its own copy, saving memory

C. Dynamic linking eliminates the need for a page table

D. Dynamically linked programs cannot have bugs

Hint

Track when an address or library reference is resolved and the direction of translation.

Answer B: Share suitable library code pages across programs.

OS27 says dynamic linking resolves external-library references at load/runtime and allows suitable library code pages to be shared, whereas static linking copies selected code into the executable. B captures the potential memory saving. It is not an unconditional speed guarantee, does not remove address translation/page tables and cannot eliminate programming bugs. Only shareable code/pages are implied, not arbitrary writable library state.

Why each choice is right or wrong

A - Incorrect: Dynamic resolution/sharing does not guarantee faster execution for every program or workload.

B - Correct: Several processes can share suitable library code pages rather than each embedding a separate static copy.

C - Incorrect: Library linking does not remove virtual-address translation or its page-table requirements.

D - Incorrect: Choosing dynamic rather than static linking cannot guarantee the absence of programming errors.

Primary note proof

Study-note section OS27

> Dynamic linking resolves references to external libraries at load/runtime, allowing suitable library code pages to be shared; static linking copies selected code into the executable.

Study-note section OS30

> Pages and physical frames have equal fixed size P. Divide virtual address v into page=floor(v/P) and offset=v mod P. The page table maps a page to a frame and protection/status flags.

Question sources

Operating Systems review questions · Question 50

Software Engineering

Questions & Worked Answers

278 questions

Question 1 | Q00003

A company delivers registration first, then adds payments and reporting, each as a usable delivery. Which model specifically names this addition-of-functionality strategy, independent of whether earlier features are also refined?

A. Waterfall

B. Incremental

C. Spiral

D. Evolutionary

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Incremental

The delivery strategy adds usable capabilities in stages so that end users receive a valuable subset before the complete product is ready. Incremental development names that addition of functionality. Evolutionary development broadly emphasizes refinement and can overlap with increments, but does not specifically name the stated delivery distinction. Spiral is organized around risk-reduction cycles, and classical waterfall normally delays usable delivery until its planned development phases are complete.

Why each choice is right or wrong

A - Incorrect: Classical waterfall organizes sequential phases rather than prioritizing staged usable subsets.

B - Correct: Incremental development adds usable capability in successive deliveries.

C - Incorrect: Spiral names risk-driven cycles; those may contain increments without defining this delivery strategy.

D - Incorrect: Evolutionary refinement can overlap with increments, but the question asks the specific name for adding usable functionality.

Primary note proof

Study-note section SE04

> Choose incremental delivery when prioritized partial capability must reach users early. Smaller changes can be easier to debug and permit feedback, though architecture and integration still require planning. Iteration need not rebuild the whole system every time, and increments need

not be perfectly independent modules. Broad descriptions of early working software can fit prototypes, iterative or Agile processes; identify the purpose of that early version.

Study-note section SE06

Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 22 | lines 8-12

This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
also provide a platform for evaluation by the user.

Software Engineering module | PDF page 20 | lines 18-22

the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
In principle, the waterfall model should only be used when the requirements are well understood
and unlikely to change radically during system development. However, the waterfall model

Question sources

MOE Computer Science Exit Exam · 2016 · Question 3

Question 2 | Q00010

If you want to test individual sub-systems of software, which phase of software testing is appropriate?

A. Integration testing

B. Unit testing

C. System testing

D. Acceptance testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: Unit testing

Unit testing tests isolated building blocks, including objects and subsystems; integration begins when individually tested components are combined.

Why each choice is right or wrong

A - Incorrect: Combines components rather than isolating a subsystem. See PDF p. 148, 144.

B - Correct: Directly tests an individual object or subsystem. See PDF p. 148, 144.

C - Incorrect: Tests components together as one system. See PDF p. 148, 144.

D - Incorrect: Checks the delivered system against the client agreement. See PDF p. 148, 144.

Primary note proof

Study-note section SE35

> Acceptance testing supports the customer/user's acceptance decision against agreed needs and criteria. Alpha typically happens in a controlled developer setting with selected users; beta uses external users in their environments. None of these levels is universally the longest. Unit is a level, while white-box is a technique; a unit can be tested using black-box cases too.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 148 | lines 19-26

> Unit testing focuses on the building blocks of the software system, that is, objects and subsystems.
> There are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall test activities, allowing us to focus on smaller units of the system. Second, unit testing makes it easier to pinpoint and correct faults, given that few components are involved in the test. Third, unit testing allows parallelism in the testing activities; that is, each component can be tested independently of one another.
> The specific candidates for unit testing are chosen from the object model and the system decomposition of the system. In principle, all the objects developed during the development

Software Engineering module | PDF page 144 | lines 17-32

> Figure 6.2 depicts an overview of the testing activities:
> • Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
> • Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
> o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
> • System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:

o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Question sources

MOE Computer Science Exit Exam · 2016 · Question 10

Question 3 | Q00019

Which characteristic of requirements specifies that the requirement should be phrased so that there is one and only one interpretation for it?

A. Unambiguous

B. Correct

C. Feasible

D. Verifiable

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: Unambiguous

A single interpretation is the unambiguous property. Correctness, feasibility and verifiability answer different questions.

Why each choice is right or wrong

A - Correct: Rules out two or more interpretations. See PDF p. 68, 70.

B - Incorrect: Accuracy to the client's view does not guarantee a unique reading. See PDF p. 68, 70.

C - Incorrect: Realism concerns implementation within constraints. See PDF p. 68, 70.

D - Incorrect: Requires a repeatable test, not a unique interpretation alone. See PDF p. 68, 70.

Primary note proof

Study-note section SE11

Correct requirements reflect actual stakeholder needs. Complete requirements address the agreed scope, normal and exceptional behavior. Consistency excludes contradictions. Clarity/unambiguity gives one intended interpretation; feasibility/realism means achievable within constraints. Verifiability means objective evidence can check fulfillment. Replace “fast for most users” with a defined metric, workload and threshold.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 68 | lines 18-26

system specification. Requirement validation involves checking if the specification is correct, complete, consistent, unambiguous, and realistic. A specification is correct if it represents the client's view of the system (i.e., everything in the requirements model accurately represents an aspect of the system). It is complete if all possible scenarios through the system are described, including exceptional behavior (i.e., all aspects of the system are represented in the requirements
model). The system specification is consistent if it does not contradict itself. The system specification is unambiguous if exactly one system is defined (i.e., it is not possible to interpret the specification two or more different ways). Finally, it is realistic if the system can be implemented within constraints. These properties are illustrated with UML instance diagrams in

Software Engineering module | PDF page 70 | lines 3-16

3.2.5. Verifiability and Traceability
Two more desirable properties of a system specification are that it be verifiable and traceable. The
specification is verifiable if, once the system is built; a repeatable test can be designed to demonstrate that the system fulfills the requirement. For example, a mean time to failure of a hundred years for SatWatch would be difficult to achieve (assuming it is realistic in the first place).
The following requirements are additional examples of non-verifiable requirements:
• The product shall have a good user interface (good is not defined).
• The product shall be error free (requires large amount of resources to establish).
• The product shall respond to the user with 1 second for most cases ("most cases" is not defined).
A system specification is traceable if each system function can be traced to its corresponding set of requirements. Traceability is not a constraint on the content of the specification, but rather, on its organization. Traceability facilitates the development of tests and the systematic validation of the design against the requirements.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 19

Computer Science Exit Exam practice collection · Question 12

Question 4 | Q00082

Which option most completely summarizes the listed observable behaviors that functional requirements may specify?

A. specifies the tasks the program must complete

B. Specifies observable tasks the program must not perform

C. Specifies required observable responses to invalid input

D. All of the mentioned

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer D: All of the mentioned

Functional requirements describe observable services and behavior, including what must happen, what is prohibited and how the system responds to errors. Each individual choice gives one of these forms. The all-of-the-mentioned option is therefore the complete summary. Selecting only A, B or C would omit other valid functional obligations; negative wording alone does not make a requirement nonfunctional.

Why each choice is right or wrong

A - Incorrect: Required tasks are functional, but A alone omits the other listed forms.

B - Incorrect: Prohibited tasks can be functional, but B alone is incomplete.

C - Incorrect: Required error responses are functional, but C alone is incomplete.

D - Correct: All three listed observable behavior forms can be functional requirements.

Primary note proof

Study-note section SE09

> A functional requirement states an observable service or behavior, including reactions to inputs, errors and forbidden actions. “When the sensor detects intrusion, send an alert” is functional. “Only a manager may approve refunds” also specifies permitted/prohibited behavior. A negative statement is not automatically nonfunctional.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 65 | lines 19-23

> 3.2.1. Functional Requirements
> Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. For example, the following is an example of functional requirements for SatWatch, a watch that resets itself without user intervention:

Question sources

MOE Computer Science Exit Exam · 2016 · Question 82

Question 5 | Q00083

A team has little experience with a project's unfamiliar technology. Major feasibility risks must be evaluated before each further commitment, and experienced risk-analysis support is available. Which lifecycle is selected specifically by that risk-driven criterion?

A. Iterative Enhancement Model

B. RAD

C. Spiral

D. Waterfall

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Spiral

Spiral organizes each cycle around objectives, alternatives, risk assessment and reduction before further commitment. That directly addresses the specified major feasibility risks and supported risk-analysis capability. Iterative enhancement can aid learning but does not make risk assessment its defining organizer; RAD needs rapid construction and skilled support. Classical waterfall fits stable requirements and familiar technology, which the scenario does not supply.

Why each choice is right or wrong

A - Incorrect: Iteration supports feedback but is not defined by risk-driven commitments.

B - Incorrect: RAD emphasizes rapid construction rather than the specified risk cycle.

C - Correct: Spiral explicitly organizes commitment around major risk reduction.

D - Incorrect: Unfamiliar technology and unresolved risks do not fit the stated classical waterfall rationale.

Primary note proof

Study-note section SE02

> Select a process using requirement stability, technical risk, delivery urgency, stakeholder availability, team skills and assurance needs. "Team has little experience" alone does not uniquely identify a process. A risk-driven cycle, prototype or carefully planned staged process may each help, depending on the actual uncertainty. No process eliminates the need for testing or documentation.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 20 | lines 18-22

> the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
> In principle, the waterfall model should only be used when the requirements are well understood
> and unlikely to change radically during system development. However, the waterfall model

Software Engineering module | PDF page 13 | lines 22-27

> Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to
> follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 83

MOE Computer Science Exit Exam · 2016 · Question 95

Question 6 | Q00084

In which step of SDLC actual programming of software code is done?

A. Development and Documentation

B. Maintenance and Evaluation

C. Design

D. Analysis

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A: Development and Documentation

Programming translates the design into source code in implementation. A corresponds to the development stage of implementation and unit testing.

Why each choice is right or wrong

A - Correct: Development is the option corresponding to producing source code; documentation accompanies the process. See PDF p. 18, 18, 19.

B - Incorrect: Maintenance modifies an operational product. See PDF p. 18, 18, 19.

C - Incorrect: Design derives implementation structure rather than source code. See PDF p. 18, 18, 19.

D - Incorrect: Analysis establishes and organizes requirements. See PDF p. 18, 18, 19.

Primary note proof

Study-note section SE02

A common lifecycle names requirements, design, implementation, testing, deployment and maintenance. Analysis clarifies the problem; system design chooses architecture; object/detail design specifies interfaces and algorithms; implementation writes code; testing evaluates behavior; deployment puts it into service. These activities may overlap and repeat.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 18 | lines 28-32

3. Implementation and unit testing: During this stage, the software design is realized as a set of programs or program units. Unit testing involves verifying that each unit meets its specification. The purpose of the coding phase (sometimes called the implementation phase) of software development is to translate the software design into source code. Each component of the design is
implemented as a program module. The end-product of this phase is a set of program modules that

Software Engineering module | PDF page 18 | lines 13-20

System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
Software design involves identifying and describing the fundamental software system abstractions
and their relationships. The goal of the design phase is to transform the requirements specified in
the SRS document into a structure that is suitable for implementation in some programming language. In technical terms, during the design phase the software architecture is derived from the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Software Engineering module | PDF page 19 | lines 24-29

5. Operation and maintenance: Normally (although not necessarily), this is the longest life cycle phase. Maintenance of a typical software product requires much more than the effort necessary to
develop the product itself. The system is installed and put into practical use. Maintenance involves
correcting errors which were not discovered in earlier stages of the life cycle, improving the

implementation of system units and enhancing the system ‘s services as new requirements are discovered.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 84

MOE Computer Science Exit Exam · 2016 · Question 96

Question 7 | Q00094

Which offered requirement is nonfunctional, specifying a measurable quality rather than required or prohibited observable tasks?

A. specifies the tasks the program must complete

B. Specifies observable tasks the program must not perform

C. Specifies the required error response to invalid input

D. Search shall respond within two seconds under 100 concurrent users.

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer D: Search shall respond within two seconds under 100 concurrent users.

The response-time limit under a defined concurrent-user workload is a measurable performance requirement, a nonfunctional quality constraint. Required services, prohibited actions and specified error responses are all forms of observable functional behavior. Their positive or negative wording does not determine the category. D instead constrains how well the search service must perform, making it the unique nonfunctional example.

Why each choice is right or wrong

A - Incorrect: Required observable tasks are functional.

B - Incorrect: Prohibited observable tasks can be functional.

C - Incorrect: An observable error response specifies functional behavior.

D - Correct: The time threshold and workload constrain performance quality.

Primary note proof

Study-note section SE09

A functional requirement states an observable service or behavior, including reactions to inputs, errors and forbidden actions. “When the sensor detects intrusion, send an alert” is functional. “Only a manager may approve refunds” also specifies permitted/prohibited behavior. A negative statement is not automatically nonfunctional.

Study-note section SE10

"Search responds within two seconds under 100 concurrent users" is measurable performance. "New users finish checkout within one minute without help" is usability. "Use Java" is a technology/pseudo-requirement, often separately classified as a system constraint. Dependability includes reliability, safety and robustness; portability is not its usual central category. Performance degradation at peak load primarily concerns performance, not automatically reliability.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 65 | lines 19-23

3.2.1. Functional Requirements
Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. For example, the following is an example of functional requirements for SatWatch, a watch that resets itself without user intervention:

Question sources

MOE Computer Science Exit Exam · 2016 · Question 94

Question 8 | Q00098

Which of the following is the understanding of software product limitations, learning system related problems or changes to be done in existing systems beforehand, identifying and addressing the impact of project on organization and personnel etc.?

A. Software Design

B. Feasibility Study

C. Requirement Gathering

D. System Analysis

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: Feasibility study.

SE08 asks whether a proposal is worth doing and deliverable, considering existing limitations, impacts on organization and personnel, benefits, costs, skills and integration risk. That matches the stem's preliminary impact and constraint investigation. System analysis studies the detailed problem and behavior and informs feasibility, but the broad early viability assessment is the more specific match here.

Why each choice is right or wrong

A - Incorrect: Design chooses solution structures rather than primarily assessing preliminary viability and organizational impacts.

B - Correct: Feasibility investigates existing limitations, impacts and whether the proposal is worthwhile and deliverable.

C - Incorrect: Gathering needs is useful input but does not by itself cover the full viability assessment.

D - Incorrect: System analysis examines detailed problem behavior; feasibility is the more specific preliminary impact/viability activity.

Primary note proof

Study-note section SE08

> Feasibility asks whether a proposal is worth doing and can be delivered within technical, economic, operational and schedule constraints. Examine existing limitations, organizational impacts, benefits, cost, available skills and integration risk. System analysis studies the problem and needed behavior in more detail; the activities inform each other.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 25 | lines 15-18

> There are four main activities in the requirements engineering process:
> 1. Feasibility study: Feasibility shows viability and practicality of a software, project or idea. An estimate is made of whether the identified user needs may be satisfied using current software and hardware technologies. The study considers whether the proposed system will be cost-

Software Engineering module | PDF page 26 | lines 3-5

> effective from a business point of view and if it can be developed within existing budgetary constraints. A feasibility study should be relatively cheap and quick. The result should inform the decision of whether or not to go ahead with a more detailed analysis.

Software Engineering module | PDF page 100 | lines 11-24

> Analysis focuses on producing a model of the system, called the analysis model, which is correct, complete, consistent, and verifiable. Analysis is different from requirements elicitation in that developers focus on structuring and formalizing the requirements elicited from users
> Figure 4.1 Products of requirements elicitation and analysis (UML activity diagram).
> Althoughtheanalysismodelmaynotbeunderstandabletotheusersandtheclient, it helps developers verify the system specification produced during requirements elicitation. There is a natural tendency for users and developers to postpone difficult decisions until later in the project. A decision may be difficult because of lack of domain knowledge, lack of technological knowledge, or simply because of disagreements among users and developers. Postponing decisions enables the
> project to move on smoothly and avoid confrontation with reality or peers. Unfortunately, difficult
> decisions will eventually need to be made, often at higher cost when intrinsic problems are discovered during testing, or worse, during user evaluation. Translating a system specification in to a formal or semi-formal model forces developers to identify and resolve difficult issues early

in
the development.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 98

Question 9 | Q00120

For a software product developed to meet a client’s requirements, which choice most completely describes its scope?

A. Software product refers any software created to meet the client's requirements.

B. Software product includes design and test documentation.

C. Software product includes requirement specifications, and user manuals.

D. All of the above

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer D: All of the above

SE01 explicitly includes executable code plus requirements, designs, test assets, configuration and user documentation in a software product. A describes the implemented software, while B and C name relevant accompanying artifacts. All three descriptions are therefore compatible; none limits the product to code alone. D is the inclusive answer among these options.

Why each choice is right or wrong

A - Incorrect: True as one component, but incomplete as the requested full selection. Software built to meet client requirements forms part of the product.

B - Incorrect: True as one component, but incomplete as the requested full selection. Relevant design and testing documentation are included product artifacts.

C - Incorrect: True as one component, but incomplete as the requested full selection. Requirements specifications and user manuals are relevant product documentation.

D - Correct: All three descriptions are compatible with the complete software-product scope.

Primary note proof

Study-note section SE01

Software engineering applies systematic methods to development, operation and maintenance. A software product includes executable code and relevant requirements, designs, test assets, configuration and user documentation. A process defines activities, ordering, roles, products and entry/exit conditions. Its four broad activities are specification, design/implementation, validation and evolution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 89 | lines 17-24

> Assigning a role to a participant usually translates into assigning the responsibility of a specific work product. A work product is an artifact that is produced during the development, such as an internal document for other project participants or a deliverable to the customer.
> An analyst is responsible for a section of the RAD describing the functionality of the system. A system architect is responsible for the SDD describing the system's high-level architecture. A project manager is responsible for the SPMP describing the development processes. Finally, the system and its accompanying documentation also constitute a set of work products that are delivered to the client.

Software Engineering module | PDF page 83 | lines 12-25

> 3.4.4. Documenting Requirements Elicitation
> The results of the requirements elicitation activity and the analysis activity are documented in the
> Requirements Analysis Document (RAD). This document completely describes the system in terms of functional and nonfunctional requirements and serves as a contractual basis between the
> client and the developers. The audience for the RAD includes the client, the users, the project management, the system analysts (i.e., the developers who participate in the requirements), and the system designers (i.e., the developers who participate in the system design). The first part of the document, including use cases and nonfunctional requirements, is written during requirements
> elicitation. The formalization of the specification in terms of object models is written during analysis.
> The first section of the RAD is an Introduction. Its purpose is to provide a brief overview of the function of the system and the reasons for its development, its scope, and references to the development context (e.g., related problem statement, references to existing systems, feasibility studies). The introduction also includes the objectives and success criteria of the project.

Software Engineering module | PDF page 130 | lines 3-11

> 6.3. DOCUMENTING SYSTEM DESIGN
> System design is documented in the System Design Document (SDD). It describes design goals set by the project, the subsystem decomposition (with UML class diagrams), the hardware/software mapping (with UML deployment diagrams), the data management, the access control, control flow mechanisms, and boundary conditions. The SDD is used to define interfaces between teams of developers and as reference when architecture-level decisions need to be revisited. The audience for the SDD includes the project management, the system architects (i.e., the developers who participate in the system design), and the developers who design and implement each subsystem. The following is an example template for a SDD:

Question sources

Addis Ababa University mock exam · Question 22

Question 10 | Q00121

One of the followings is type of requirements associated with quality features of a software.

A. Business requirements

B. Functional requirements requirements

C. Non-functional

D. User requirements

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Non-functional

Quality and user-visible constraints beyond functional behavior are nonfunctional requirements; user and business are viewpoints rather than this quality classification.

Why each choice is right or wrong

A - Incorrect: Business objectives do not identify the stated quality category. See PDF p. 66, 78.

B - Incorrect: Specifies provided interactions and services. See PDF p. 66, 78.

C - Correct: Matches nonfunctional quality constraints. See PDF p. 66, 78.

D - Incorrect: User requirements can contain several types, so this is broader than quality features. See PDF p. 66, 78.

Primary note proof

Study-note section SE10

> Nonfunctional requirements include quality attributes and constraints. Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning. Security concerns protection; compatibility concerns cooperating environments; portability concerns moving environments; maintainability/supportability concern change, diagnosis and evolution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 66 | lines 14-18

> 3.2.2. Non-Functional and Pseudo-Requirements
> Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Nonfunctional requirements include quantitative constraints, such as response time (i.e., how fast the system reacts to user commands)
> or accuracy (i.e., how precise are the system’s numerical answers). The following are the

Software Engineering module | PDF page 78 | lines 12-30

3.3.7. Identifying Non-Functional Requirements
Nonfunctional requirements describe user-visible aspects of the system that are not directly related
to the functional behavior of the system. Nonfunctional requirements span a number of issues, from user interface look and feel to response time requirements to security issues. Nonfunctional
requirements are defined at the same time as functional requirements are, because they have as much impact on the development and cost of the system.
For example, consider a mosaic display that an air traffic controller uses to track planes. A mosaic
display system complies data from a series of radar and databases (hence the term “mosaic”) into
a summary display indicating all aircraft in a certain area, including their identification, speed, and
altitude. The number of aircraft such a system can display constrains the performance of the air traffic controller and the cost of the system. If the system can only handle a few aircraft simultaneously, the system cannot be used at busy airports. On the other hand, a system able to handle a large number of aircraft is more costly and more complex to build.
Nonfunctional requirements can be elicited by investigating the following issues.
• User interface and human factors. What kind of interface should the system provide? What is the level of expertise of the users?
• Documentation. What level of document is required? Should only user documentation be provided? Should there be technical documentation for maintainers? Should the development process be documented?

Question sources

Addis Ababa University mock exam · Question 23

Question 11 | Q00122

A type of requirement that describes why the software project is needed and used to define the objective of the software is .

A. Business requirements

B. Functional requirements

C. Non-functional requirements

D. User requirements

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: Business requirements.

SE08 distinguishes the business reason and objectives of a project from user needs and detailed system obligations. The question asks why the project is needed and what objectives it serves, so business requirements match directly. Functional requirements describe behavior, while nonfunctional requirements describe qualities or constraints; neither primarily states the project's business justification.

Why each choice is right or wrong

A - Correct: Business requirements state the project's purpose, justification and intended organizational goals.

B - Incorrect: Functional requirements specify observable system services rather than the overall business reason for the project.

C - Incorrect: Nonfunctional requirements specify qualities or constraints rather than why the project exists.

D - Incorrect: User requirements describe stakeholder needs at a high level rather than the project's business justification.

Primary note proof

Study-note section SE08

> Business requirements explain why the project exists and its goals. User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints. The Software Requirements Specification (SRS) organizes the agreed requirements. A natural-language specification communicates with clients while analysis models formalize it for technical discussion; real specifications can also contain diagrams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 83 | lines 12-25

> 3.4.4. Documenting Requirements Elicitation
> The results of the requirements elicitation activity and the analysis activity are documented in the
> Requirements Analysis Document (RAD). This document completely describes the system in terms of functional and nonfunctional requirements and serves as a contractual basis between the
> client and the developers. The audience for the RAD includes the client, the users, the project management, the system analysts (i.e., the developers who participate in the requirements), and the system designers (i.e., the developers who participate in the system design). The first part of the document, including use cases and nonfunctional requirements, is written during requirements
> elicitation. The formalization of the specification in terms of object models is written during analysis.
> The first section of the RAD is an Introduction. Its purpose is to provide a brief overview of the function of the system and the reasons for its development, its scope, and references to the

development context (e.g., related problem statement, references to existing systems, feasibility studies). The introduction also includes the objectives and success criteria of the project.

Software Engineering module | PDF page 26 | lines 10-17

3. Requirement's specification: Requirements specification is the activity of translating the information gathered during the analysis activity in to a document that defines a set of requirements. Two types of requirements may be included in this document. User requirements are
abstract statements of the system requirements for the customer and end-user of the system; system
requirements are a more detailed description of the functionality to be provided.
4. Requirement's validation: This activity checks the requirements for realism, consistency, and completeness. During this process, errors in the requirements document are inevitably discovered.
It must then be modified to correct these problems.

Question sources

Addis Ababa University mock exam · Question 24

Question 12 | Q00123

Which of the following is used to model the functionality of software?

A. Use case diagram

B. Class diagram

C. E-R diagram

D. Deployment diagram

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer A: Use case diagram

Use-case diagrams model external roles and their goals or interactions with the system. This is the functional view requested. Class diagrams instead emphasize static classes, features and relationships. Entity-relationship diagrams organize entity types, attributes and data relationships. Deployment diagrams map software artifacts to execution nodes. Each can contribute to understanding a system, but only the use-case view directly presents the actor-goal functionality in this comparison.

Why each choice is right or wrong

A - Correct: Use-case diagrams show external actors and the system goals they use.

B - Incorrect: Class diagrams describe static types and their relationships.

C - Incorrect: ER diagrams describe entity types, attributes and data relationships.

D - Incorrect: Deployment diagrams map software artifacts onto execution nodes.

Primary note proof

Study-note section SE14

> Draw actors outside the system boundary rectangle and use-case ovals inside. Name a goal with a verb and noun. The diagram shows roles, goals and relationships, not the full detailed algorithm or UI layout. Being an internal component is different from being an external actor; the chosen boundary determines which role applies.

Study-note section SE-S03

> An entity-relationship diagram models entity types, their attributes and the relationships between them, commonly for data design. Its focus differs from a use-case diagram of external roles and goals, a class diagram of classes and their features, or a deployment diagram mapping artifacts to execution nodes.

Study-note section SE24

> A component diagram shows components/interfaces/dependencies; a deployment diagram places software artifacts on nodes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 47 | lines 30-39

> Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
> Use cases describe the behavior of the system as seen from an actor’s point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate a
> use case to access the system functionality. The use case can then initiate other use cases and gather
> more information from the actors. When actors and use cases exchange information, they are said
> to communicate. We will see later that we represent these exchanges with communication relationships.

Software Engineering module | PDF page 51 | lines 14-24

> 2.3.2. Class Diagrams
> Classes and objects
> Class diagrams describe the structure of the system in terms of classes and objects. Classes are abstractions that specify the attributes and behavior of a set of objects. Objects are entities that encapsulate state and behavior. Each object has an identity: It can be referred individually and is distinguishable from other objects.
> In UML, classes and objects are depicted by boxes including three compartments. The top

compartment displays the name of the class or object. The center compartment displays its attributes, and the bottom compartment displays its operations. The attribute and operation compartment can be omitted for clarity. Object names are underlined to indicate that they are instances. By convention, class names start with an uppercase letter. Objects in object diagrams

Software Engineering module | PDF page 130 | lines 3-11

6.3. DOCUMENTING SYSTEM DESIGN
System design is documented in the System Design Document (SDD). It describes design goals set by the project, the subsystem decomposition (with UML class diagrams), the hardware/software mapping (with UML deployment diagrams), the data management, the access control, control flow mechanisms, and boundary conditions. The SDD is used to define interfaces between teams of developers and as reference when architecture-level decisions need to be revisited. The audience for the SDD includes the project management, the system architects (i.e., the developers who participate in the system design), and the developers who design and implement each subsystem. The following is an example template for a SDD:

Question sources

Addis Ababa University mock exam · Question 25

Question 13 | Q00124

Which of the following refers to the deployment platform? i

A. Use case

B. Tier

C. Layer

D. Component

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: Tier.

SE23 explicitly separates logical abstraction from deployment separation. A layer organizes responsibilities logically, whereas a tier describes their deployment separation or platform arrangement. A use case models an actor goal, and a component is a software building block. Neither is the term for the deployment distinction asked for here.

Why each choice is right or wrong

A - Incorrect: A use case models a goal or interaction rather than the deployment platform separation.

B - Correct: A tier identifies deployment separation, contrasting with logical layering.

C - Incorrect: A layer is a logical abstraction level and need not occupy a distinct deployment platform.

D - Incorrect: A component is a software element and does not itself denote a deployment tier.

Primary note proof

Study-note section SE23

A layer is a logical abstraction level; a tier is a deployment separation. A closed layered architecture permits calls only to the immediately lower layer; an open one permits approved skipping. For top-to-bottom A,B,C,D, A→B obeys closed layering, A→C skips, and lower-to-upper calls violate the simple dependency rule unless explicitly mediated by a defined mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 130 | lines 3-11

6.3. DOCUMENTING SYSTEM DESIGN
System design is documented in the System Design Document (SDD). It describes design goals set by the project, the subsystem decomposition (with UML class diagrams), the hardware/software mapping (with UML deployment diagrams), the data management, the access control, control flow mechanisms, and boundary conditions. The SDD is used to define interfaces between teams of developers and as reference when architecture-level decisions need to be revisited. The audience for the SDD includes the project management, the system architects (i.e., the developers who participate in the system design), and the developers who design and implement each subsystem. The following is an example template for a SDD:

Software Engineering module | PDF page 67 | lines 3-8

Pseudo-requirements are requirements imposed by the client that restrict the implementation of the system. Typical pseudo-requirements are the implementation language and the platform on which the system is to be implemented. For life-critical developments, pseudo-requirements often
include process and documentation requirements (e.g., the use of a formal specification method, the complete release of all work products). Pseudo-requirements have usually no direct effect on the users' view of the system. The following are the pseudo functional requirements for SatWatch:

Question sources

Addis Ababa University mock exam · Question 26

Question 14 | Q00125

When requirements will change substantially and frequent early feedback is needed, which classical process organization is least accommodating among these choices?

A. Waterfall

B. Scrum

C. RAD

D. Agile

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A: Waterfall

Classical waterfall commits to planned phases and reviewed deliverables, making late feedback and changed requirements costly. Agile and Scrum plan and inspect work incrementally, allowing later content to adapt. RAD also uses rapid construction and user participation to obtain feedback. The comparison therefore favors waterfall as the least accommodating under the stated conditions, without claiming that a waterfall project is physically unable to authorize any change.

Why each choice is right or wrong

A - Correct: Late feedback and phase commitments make major requirement changes costly in classical waterfall.

B - Incorrect: Scrum inspects usable increments and adapts later work within its planning framework.

C - Incorrect: RAD uses rapid construction and user participation to obtain feedback.

D - Incorrect: Agile explicitly values responding to changing needs.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Study-note section SE05

> Agile favors people collaborating, useful working results, customer participation and adaptation over rigid adherence to procedures, exhaustive documents, contracts or fixed plans. The latter still have value. Agile does not mean no planning or no documentation. A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Study-note section SE07

A prototype explores requirements, usability or technical feasibility. A throwaway prototype is discarded after learning; an evolutionary prototype is deliberately refined toward production. A quick prototype may omit robustness, security or architecture, so do not deploy it unchanged without evaluation. Coding may occur during prototyping even if a diagram labels that activity construction rather than coding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 20 | lines 18-22

the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
In principle, the waterfall model should only be used when the requirements are well understood
and unlikely to change radically during system development. However, the waterfall model

Software Engineering module | PDF page 13 | lines 22-27

Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to
follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Question sources

Addis Ababa University mock exam · Question 27

Question 15 | Q00126

Which of the following aims to find errors?

A. Requirement Analysis

B. Design

C. Testing

D. None of the above

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer C: Testing

Testing systematically attempts to expose errors in the implementation; the other named activities primarily construct specifications and designs.

Why each choice is right or wrong

A - Incorrect: Structures requirements, although validation may reveal requirement defects. See PDF p. 140, 100, 18.

B - Incorrect: Constructs the solution design. See PDF p. 140, 100, 18.

C - Correct: Explicit planned fault detection activity. See PDF p. 140, 100, 18.

D - Incorrect: Testing is present, so none is false. See PDF p. 140, 100, 18.

Primary note proof

Study-note section SE34

A fault/defect is an underlying problem; an error may be an incorrect internal state; a failure is an externally observed departure from required service. Reliability is a product quality attribute; low cost or short development time are project outcomes. Tests increase confidence and reveal defects but cannot generally guarantee correctness over every possible input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 140 | lines 27-32

7.1 OVERVIEW OF SOFTWARE QUALITY ASSURANCE
We define testing as the systematic attempt to find errors in a planned way in the implemented software. Contrast this definition with another commonly used definition that says that “testing is
the process of demonstrating that errors are not present.” The distinction between these two definitions is important. Our definition does not mean we simply demonstrate that the program does what it is intended to do. The explicit goal of testing is to demonstrate the presence of faults.

Software Engineering module | PDF page 100 | lines 11-24

Analysis focuses on producing a model of the system, called the analysis model, which is correct, complete, consistent, and verifiable. Analysis is different from requirements elicitation in that developers focus on structuring and formalizing the requirements elicited from users
Figure 4.1 Products of requirements elicitation and analysis (UML activity diagram).
Althoughtheanalysismodelmaynotbeunderstandabletotheusersandtheclient, it helps developers verify the system specification produced during requirements elicitation. There is a natural tendency for users and developers to postpone difficult decisions until later in the project. A decision may be difficult because of lack of domain knowledge, lack of technological knowledge, or simply because of disagreements among users and developers. Postponing decisions enables the
project to move on smoothly and avoid confrontation with reality or peers. Unfortunately, difficult
decisions will eventually need to be made, often at higher cost when intrinsic problems are discovered during testing, or worse, during user evaluation. Translating a system specification in to a formal or semi-formal model forces developers to identify and resolve difficult issues early in
the development.

Software Engineering module | PDF page 18 | lines 13-20

System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
Software design involves identifying and describing the fundamental software system abstractions
and their relationships. The goal of the design phase is to transform the requirements specified in
the SRS document into a structure that is suitable for implementation in some programming language. In technical terms, during the design phase the software architecture is derived from the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Question sources

Addis Ababa University mock exam · Question 28

Question 16 | Q00239

Which among the following focuses on the way the elements interact with each other at runtime during architectural design?

A. All choices are correct

B. Component-and-connector (C&C) structures

C. Allocation structures

D. Module decomposition structure

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: Component-and-connector structures.

The runtime interaction of software elements is the subject of component-and-connector structures in SE23. Module structures instead organize the code, while allocation structures relate software to hardware, teams or environments. These are different architectural views, so the inclusive choice fails: only the C&C option specifically answers the runtime-interaction focus.

Why each choice is right or wrong

A - Incorrect: The alternatives address different architectural concerns, so not every choice focuses on runtime interaction.

B - Correct: C&C structures show runtime elements and the connectors through which they interact.

C - Incorrect: Allocation structures map software to hardware, teams or environments rather than primarily describing runtime exchanges.

D - Incorrect: Module decomposition shows code organization rather than the runtime interactions emphasized by the stem.

Primary note proof

Study-note section SE23

A structure is the elements/relations themselves; a view represents selected aspects. Module structures show code organization, component-and-connector structures runtime interaction, and allocation structures map software to hardware, teams or environment. Processes/threads belong in concurrency views. Evaluate architectures against quality attributes and scenarios; processor allocation, decomposition and style selection can all be architectural concerns.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 121 | lines 6-15

6.1.4. Software architecture
As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept
of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
In this section, we describe a few sample architectures that can be used for different systems. This
is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
representative examples and refer the reader to the literature for more details.

Question sources

Mock exam 4 · Question 41

Question 17 | Q00240

Which option combines both explicitly specified testability mechanisms among A, C and D? Treat refactoring alone as behavior-preserving restructuring with no stated testability improvement.

A. Sand box

B. A and D

C. Refactoring

D. Limiting structural complexity

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: A and D

A sandbox can isolate execution, giving control over the test environment. Limiting structural complexity also supports testability by making behavior easier to control and observe. Refactoring preserves behavior while changing structure, but without a stated improvement it does not automatically supply either testability effect. The combination A and D therefore includes both explicitly supported mechanisms; the individual choices each give only one.

Why each choice is right or wrong

A - Incorrect: Sandbox isolation is valid but only one of the two mechanisms.

B - Correct: A and D include both explicitly supported mechanisms.

C - Incorrect: Unspecified refactoring need not improve testability.

D - Incorrect: Limited structural complexity is valid but only one of the two mechanisms.

Primary note proof

Study-note section SE27

> Variability can use selectable features, omission/inclusion and extension points; overloading supplies alternate signatures in code but is not every variability mechanism. Testability improves with controllable inputs, observable outputs, replaceable dependencies and limited structural complexity. A sandbox can isolate execution. Assess each design claim independently before evaluating a combined statement.

Study-note section SE22

> Logical cohesion groups similar-category operations, often selected by a flag; temporal cohesion groups operations done at the same time; functional cohesion focuses on one task. Splitting an unrelated three-purpose component improves modularity. If behavior is preserved, the restructuring is refactoring; the words are not mutually exclusive answers. Smaller modules can aid understanding, independent work, reuse and maintenance, but too many interfaces also create complexity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 127 | lines 10-28

> 6.2.1. Identifying design goals
> The definition of design goals is the first step of system design. It identifies the qualities that our system should focus on. Many design goals can be inferred from the nonfunctional requirements or from the application domain. Others will have to be elicited from the client. It is, however, necessary to state them explicitly such that every important design decision can be made consistently following the same set of criteria.
> For example, in the light of the nonfunctional requirements for MyTrip, we identify reliability and
> fault tolerance to connectivity loss as design goals. We then identify security as a design goal, as numerous drivers will have access to the same trip planning server. We add modifiability as a design goal, as we want to provide the ability for drivers to select a trip planning service of their

choice. The following box summarizes the design goals we identified.
In general, we can select design goals from a long list of highly desirable qualities. Tables 5-1-5-5 list a number of possible design criteria. These criteria are organized into five groups: performance, dependability, cost, maintenance, and end user criteria. Performance, dependability,
and end user criteria are usually specified in the requirements or inferred from the application domain. Cost and maintenance criteria are dictated by the customer and the supplier.
Performance criteria (Table 5-1) include the speed and space requirements imposed on the system. Should the system be responsive or should it accomplish a maximum number of tasks? Is memory space available for speed optimizations or should memory be used sparingly?

Software Engineering module | PDF page 133 | lines 3-15

transformation on subsets of a particular model. Typical activities include transforming n-ary associations into binary associations, implementing binary associations as references, merging two similar classes from two different subsystems into a single class, collapsing classes with no significant behavior into attributes, splitting complex classes into simpler ones, rearranging classes and operations to increase the inheritance and packaging. During restructuring, we address design goals such as maintainability, readability, and understandability of the system model.
• Optimization. Optimization activities address performance requirements of the system model. This includes changing algorithms to respond to speed or memory requirements, reducing multiplicities in associations to speed up queries, adding redundant associations for efficiency, rearranging execution orders, adding derived attributes to improve the access time to objects and opening up the architecture, that is, adding access to lower layers because of performance requirements.

Question sources

Mock exam 4 · Question 42

Question 18 | Q00241

Which choices explicitly provide architectural variability through selectable elements or defined extension interfaces? Select all that apply. Treat overloading as alternate code signatures only.

A. Overloading

B. Omission or inclusion of elements

C. Extension point

D. All of the above

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B, C: Omission or inclusion of elements; Extension point

Omitting or including selected elements permits configured architectural variation, and extension points define places where additional behavior can be supplied. Overloading by itself supplies alternate code signatures and does not establish either architectural mechanism. Thus B and C meet the specified architectural scope. The all-of-the-above choice includes that insufficient code-level case and therefore cannot be selected as a correct aggregate.

Why each choice is right or wrong

A - Incorrect: Alternate signatures alone do not define the specified architectural variability.

B - Correct: Selectable omission or inclusion changes architectural configuration.

C - Correct: Extension points explicitly support variation at defined boundaries.

D - Incorrect: It incorrectly includes overloading alone as an architectural mechanism.

Primary note proof

Study-note section SE27

> Variability can use selectable features, omission/inclusion and extension points; overloading supplies alternate signatures in code but is not every variability mechanism. Testability improves with controllable inputs, observable outputs, replaceable dependencies and limited structural complexity. A sandbox can isolate execution. Assess each design claim independently before evaluating a combined statement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 121 | lines 6-15

> 6.1.4. Software architecture
> As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept
> of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
> In this section, we describe a few sample architectures that can be used for different systems. This
> is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
> representative examples and refer the reader to the literature for more details.

Software Engineering module | PDF page 104 | lines 6-17

> 5.1.4. Generalization
> Generalization enables us to organize concepts in to hierarchies. There may be any number of intermediate levels in between, covering more-or-less generalized concepts (e.g., Low Priority Incident, Emergency, Disaster). Such hierarchies allow us to refer to many concepts precisely.

When
we use the term Incident, we mean all instances of all types of Incidents. When we use the term Emergency, we only refer to an Incident that requires an immediate response. This view of generalization stems from modeling.
In an object-oriented programming language, inheritance is a reusability technique. If a class Child
inherits from a class Parent, all the attributes and methods available on Parent are automatically available on Child. The class Child may add additional methods or override inherited methods, thus refining the class Parent. As in the case of generalization, the class at the top of the hierarchy
tends to be the most general one, whereas the leaves tend to be the most specialized.

Question sources

Mock exam 4 · Question 43

Question 19 | Q00242

Which choice most completely describes the stated advantages of the Factory Method pattern?

A. Allows the sub-classes to choose the type of objects to create

B. Promotes loose coupling

C. A and B

D. None of the above

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: A and B

SE27 describes Factory Method as letting subclasses choose a concrete product through a creation operation, reducing dependence on concrete constructors. Those are exactly the two benefits in A and B, so C is the inclusive answer. The reference is to Factory Method; a simple factory is related but does not necessarily delegate product choice to subclasses.

Why each choice is right or wrong

A - Incorrect: True as one component, but incomplete as the requested full selection. Factory Method lets subclasses select the concrete product through the creation operation.

B - Incorrect: True as one component, but incomplete as the requested full selection. Depending on the creation abstraction reduces coupling to concrete product constructors.

C - Correct: Both listed benefits hold for Factory Method, making this the inclusive answer.

D - Incorrect: The note explicitly supports both benefits, so none is false.

Primary note proof

Study-note section SE27

Factory Method lets subclasses choose a concrete product through a creation operation, reducing coupling to concrete constructors. A simple factory centralizes creation but is not identical to Factory Method. Singleton controls access to one instance; Observer notifies subscribers; Memento stores/restores state while hiding representation; Iterator traverses; Decorator adds behavior through wrapping. The pattern name is Memento, not Momento.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 120 | lines 6-12

6.1.3. Coupling and coherence
Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Software Engineering module | PDF page 104 | lines 6-17

5.1.4. Generalization
Generalization enables us to organize concepts in to hierarchies. There may be any number of intermediate levels in between, covering more-or-less generalized concepts (e.g., Low Priority Incident, Emergency, Disaster). Such hierarchies allow us to refer to many concepts precisely. When
we use the term Incident, we mean all instances of all types of Incidents. When we use the term Emergency, we only refer to an Incident that requires an immediate response. This view of generalization stems from modeling.
In an object-oriented programming language, inheritance is a reusability technique. If a class Child
inherits from a class Parent, all the attributes and methods available on Parent are automatically available on Child. The class Child may add additional methods or override inherited methods, thus refining the class Parent. As in the case of generalization, the class at the top of the hierarchy
tends to be the most general one, whereas the leaves tend to be the most specialized.

Question sources

Mock exam 4 · Question 44

Question 20 | Q00243

Which statements correctly distinguish architectural structures and views? Select all that apply.

A. B and D only

B. Views design structures whereas architects document the structures.

C. A structure is the set of elements and their relations as they exist in software or hardware.

D. A view is a representation of a structure

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C, D: A structure is the set of elements and their relations as they exist in software or hardware.; A view is a representation of a structure

An architectural structure consists of elements and their relations in the modeled software or hardware. A view represents selected aspects of a structure for a particular purpose. Both C and D therefore express compatible definitions. Views are representations, not agents that design structures, so B is false. The aggregate A includes that false statement and incorrectly excludes the true structure definition.

Why each choice is right or wrong

A - Incorrect: The B-and-D-only combination includes false B and excludes true C.

B - Incorrect: Views represent structures rather than acting as their designers.

C - Correct: A structure is the underlying set of elements and relations.

D - Correct: A view represents selected aspects of a structure.

Primary note proof

Study-note section SE23

> A structure is the elements/relations themselves; a view represents selected aspects. Module structures show code organization, component-and-connector structures runtime interaction, and allocation structures map software to hardware, teams or environment. Processes/threads belong in concurrency views. Evaluate architectures against quality attributes and scenarios; processor allocation, decomposition and style selection can all be architectural concerns.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 121 | lines 6-15

> 6.1.4. Software architecture
> As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept

of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
In this section, we describe a few sample architectures that can be used for different systems. This
is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
representative examples and refer the reader to the literature for more details.

Software Engineering module | PDF page 130 | lines 3-11

6.3. DOCUMENTING SYSTEM DESIGN
System design is documented in the System Design Document (SDD). It describes design goals set by the project, the subsystem decomposition (with UML class diagrams), the hardware/software mapping (with UML deployment diagrams), the data management, the access control, control flow mechanisms, and boundary conditions. The SDD is used to define interfaces between teams of developers and as reference when architecture-level decisions need to be revisited. The audience for the SDD includes the project management, the system architects (i.e., the developers who participate in the system design), and the developers who design and implement each subsystem. The following is an example template for a SDD:

Question sources

Mock exam 4 · Question 45

Question 21 | Q00244

Which testing level isolates one function or class, using stubs or mocks for called dependencies, so its internal branches can be tested directly?

A. System testing

B. Integration testing

C. System and integration testing together

D. Unit testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer D: Unit testing

Unit testing isolates a function, class or small component and may replace dependencies with stubs or mocks. That matches the stated test scope. White-box cases can then target its internal branches, but white-box is a technique rather than a testing level. Integration and system tests cover combined interactions or the full system, so neither matches the isolation specified here.

Why each choice is right or wrong

A - Incorrect: System testing evaluates the complete integrated system.

B - Incorrect: Integration testing targets interactions among combined parts.

C - Incorrect: Combining system and integration levels does not give isolated unit scope.

D - Correct: Unit testing isolates the function or class with replaceable dependencies.

Primary note proof

Study-note section SE35

> Unit testing isolates a function, class or small component, often replacing dependencies with stubs/mocks; a driver invokes the unit and a stub substitutes a called dependency. Integration testing exercises interfaces and interactions among combined parts. System testing checks the complete integrated system against functional and nonfunctional requirements in a representative environment.

Study-note section SE36

> Functional tests check required behavior against the SRS and acceptance criteria. Performance tests assess speed, throughput and resources under a specified workload; load uses expected demands and stress exceeds normal limits. Usability, security, compatibility and reliability tests target their respective qualities. “Normal operating conditions” alone does not distinguish functional from performance testing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 149 | lines 24-29

> Path testing
> Path testing is a white box testing technique that identifies faults in the implementation of the component. The assumption behind path testing is that, by exercising all possible paths through the code at least once, most faults will trigger failures. The identification of paths requires knowledge of the source code and data structures.
> The starting point for path testing is the flow graph. A flow graph consists of nodes representing

Software Engineering module | PDF page 148 | lines 19-26

> Unit testing focuses on the building blocks of the software system, that is, objects and subsystems.
> There are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall test activities, allowing us to focus on smaller units of the system. Second, unit testing makes it easier to pinpoint and correct faults, given that few components are involved in the test. Third, unit testing allows parallelism in the testing activities; that is, each component can be tested independently of one another.
> The specific candidates for unit testing are chosen from the object model and the system decomposition of the system. In principle, all the objects developed during the development

Question sources

Mock exam 4 · Question 46

Question 22 | Q00245

Which offered UML diagram models external behavioral goals and actor interactions at a high level, rather than detailed message timing or state evolution?

A. Component diagram

B. Use case diagram

C. Deployment diagram

D. Class Diagram

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: Use case diagram

A use-case diagram presents external roles, their goals and relationships within a chosen system boundary. It therefore matches the specified high-level behavioral scope. Component, deployment and class diagrams describe structural organization, allocation or static types. Sequence, activity and state diagrams would be appropriate for detailed interactions, workflows or state changes, but those finer-grained dynamic questions are explicitly outside this stem.

Why each choice is right or wrong

A - Incorrect: Component diagrams describe components and interfaces.

B - Correct: Use-case diagrams show actor roles and high-level goals.

C - Incorrect: Deployment diagrams allocate artifacts to nodes.

D - Incorrect: Class diagrams show static types and relationships.

Primary note proof

Study-note section SE20

> An activity diagram models workflow: filled circle start, rounded actions, guarded decision diamond, thick fork/join bars and partitions/swimlanes for responsibility. A diamond can also merge alternative flows; a join synchronizes concurrent flows. A state machine models states and transitions triggered by events, often with guards/actions, for one object's lifecycle.

Study-note section SE14

> Draw actors outside the system boundary rectangle and use-case ovals inside. Name a goal with a verb and noun. The diagram shows roles, goals and relationships, not the full detailed algorithm or UI layout. Being an internal component is different from being an external actor; the chosen boundary determines which role applies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 101 | lines 5-6

> In UML, the functional model is represented with use case diagrams, the object model with class diagrams, and the dynamic model with statechart and sequence diagrams.

Software Engineering module | PDF page 47 | lines 30-39

> Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
> Use cases describe the behavior of the system as seen from an actor's point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate a
> use case to access the system functionality. The use case can then initiate other use cases and gather
> more information from the actors. When actors and use cases exchange information, they are said
> to communicate. We will see later that we represent these exchanges with communication relationships.

Question sources

Mock exam 4 · Question 47

Question 23 | Q00246

Which choice most completely identifies the listed reasons requirements elicitation can be difficult?

A. All choices are correct

B. Problem of scope

C. Problem of understanding

D. Problem of volatility

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: All choices are correct

SE11 identifies scope, understanding and volatility as distinct elicitation difficulties. Scope concerns agreeing what belongs in the system; understanding concerns discovering and interpreting actual needs; volatility concerns needs changing as the project progresses. Since all three listed problems are valid, the inclusive answer is appropriate rather than choosing just one source of difficulty.

Why each choice is right or wrong

A - Correct: All three listed difficulties are explicitly recognized in the complete note.

B - Incorrect: True as one component, but incomplete as the requested full selection. Unclear system boundaries make agreeing on included and excluded requirements difficult.

C - Incorrect: True as one component, but incomplete as the requested full selection. Different interpretations and tacit stakeholder needs complicate accurate understanding.

D - Incorrect: True as one component, but incomplete as the requested full selection. Changing needs make elicitation an ongoing activity rather than a one-time capture.

Primary note proof

Study-note section SE11

> Correct requirements reflect actual stakeholder needs. Complete requirements address the agreed scope, normal and exceptional behavior. Consistency excludes contradictions. Clarity/unambiguity gives one intended interpretation; feasibility/realism means achievable within constraints. Verifiability means objective evidence can check fulfillment. Replace “fast for most users” with a defined metric, workload and threshold.

Study-note section SE11

> Trace each requirement backward to its source/rationale and forward to design, code and tests. Record dependencies among requirements. A traceability matrix links these artifacts and helps impact analysis; it is not just a cost/schedule ledger. Stakeholders must negotiate conflicting requirements rather than independently approving incompatible demands. Scope, understanding and volatility are distinct elicitation difficulties.

Study-note section SE-S01

> Elicitation difficulties have several distinct sources. Scope problems concern agreeing on system boundaries and the appropriate level of detail. Understanding problems arise when stakeholders and analysts have different domain knowledge, omit tacit needs or communicate conflicting interpretations. Volatility means that requirements change as circumstances and understanding develop.

Study-note section SE-S01

> These difficulties call for repeated communication, explicit boundaries, negotiated meanings and controlled revision. They can occur together; resolving one does not automatically remove the others.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 26 | lines 6-9

> 2. Requirement’s elicitation and analysis: This is the process of deriving the system requirements through observation of existing systems, discussions with potential users and procurers, task analysis, and so on. This may involve the development of one or more system models and prototypes. These help you understand the system to be specified.

Software Engineering module | PDF page 68 | lines 18-26

system specification. Requirement validation involves checking if the specification is correct, complete, consistent, unambiguous, and realistic. A specification is correct if it represents the client's view of the system (i.e., everything in the requirements model accurately represents an aspect of the system). It is complete if all possible scenarios through the system are described, including exceptional behavior (i.e., all aspects of the system are represented in the requirements
model). The system specification is consistent if it does not contradict itself. The system specification is unambiguous if exactly one system is defined (i.e., it is not possible to interpret the specification two or more different ways). Finally, it is realistic if the system can be implemented within constraints. These properties are illustrated with UML instance diagrams in

Software Engineering module | PDF page 41 | lines 16-17

✓ Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous and the software must evolve to remain useful.

Question sources

Mock exam 4 · Question 48

Question 24 | Q00247

Which development activity primarily recognizes and clarifies the problem, required behavior and related constraints before choosing the solution architecture or algorithms?

A. Implementation

B. Testing

C. Analysis

D. Design

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Analysis

Analysis clarifies the problem, required behavior and constraints. Design then chooses architecture, interfaces and algorithms for a solution; implementation writes it, and testing evaluates its behavior. The revised scope keeps these responsibilities distinct instead of mixing problem recognition with selecting methods to overcome it. Creativity can occur in several activities, so the activity's purpose, rather than that adjective, determines the answer.

Why each choice is right or wrong

A - Incorrect: Implementation writes the selected solution.

B - Incorrect: Testing evaluates behavior.

C - Correct: Analysis clarifies the problem and constraints.

D - Incorrect: Design selects the solution architecture and mechanisms.

Primary note proof

Study-note section SE02

A common lifecycle names requirements, design, implementation, testing, deployment and maintenance. Analysis clarifies the problem; system design chooses architecture; object/detail design specifies interfaces and algorithms; implementation writes code; testing evaluates behavior; deployment puts it into service. These activities may overlap and repeat.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 100 | lines 11-24

Analysis focuses on producing a model of the system, called the analysis model, which is correct, complete, consistent, and verifiable. Analysis is different from requirements elicitation in that developers focus on structuring and formalizing the requirements elicited from users
Figure 4.1 Products of requirements elicitation and analysis (UML activity diagram).
Althoughtheanalysismodelmaynotbeunderstandabletotheusersandtheclient, it helps developers verify the system specification produced during requirements elicitation. There is a natural tendency for users and developers to postpone difficult decisions until later in the project. A decision may be difficult because of lack of domain knowledge, lack of technological knowledge, or simply because of disagreements among users and developers. Postponing decisions enables the
project to move on smoothly and avoid confrontation with reality or peers. Unfortunately, difficult
decisions will eventually need to be made, often at higher cost when intrinsic problems are discovered during testing, or worse, during user evaluation. Translating a system specification in to a formal or semi-formal model forces developers to identify and resolve difficult issues early in
the development.

Software Engineering module | PDF page 18 | lines 13-20

System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
Software design involves identifying and describing the fundamental software system abstractions
and their relationships. The goal of the design phase is to transform the requirements specified in
the SRS document into a structure that is suitable for implementation in some programming language. In technical terms, during the design phase the software architecture is derived from the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Question sources

Mock exam 4 · Question 49

Question 25 | Q00248

_____ is a set of notations and conventions used to describe and model an application.

A. All choices are correct

B. Unified approach (UA)

C. Unified modeling language (UML)

D. Jacobson's use case

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Unified modeling language (UML)

UML provides modeling notations and conventions; it is distinct from a development methodology.

Why each choice is right or wrong

A - Incorrect: UA and use cases are not both equivalent to the language. See PDF p. 32.

B - Incorrect: UA specifies steps for development. See PDF p. 32.

C - Correct: The exact defined modeling notation language. See PDF p. 32.

D - Incorrect: A use case is an interaction model inside the approach. See PDF p. 32.

Primary note proof

Study-note section SE17

A class box lists name, attributes and operations; + public, - private, # protected and ~ package indicate visibility. A class diagram shows static types and relationships, while an object diagram shows particular instances/links. UML is maintained and standardized by OMG; it is a modeling language, not a lifecycle process. Structural things describe conceptual or physical structural elements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 32 | lines 22-27

1.3.2. Overview of the unified approach.
The unified modeling language (UML) is a set of notations and conventions used to describe and model an application. But, the UML does not specify a methodology or what steps to follow to develop an application; that would be the task of the UA. Figure 1.2 depicts the essence of the unified approach. The heart of the UA is Jacobson's use case. The use case represents a typical interaction between a user and a computer system to capture the users' goals and needs. The main

Question sources

Mock exam 4 · Question 50

Question 26 | Q00249

Which structural risk can arise when repeated software changes are made without coherent design or refactoring?

A. Reduced user engagement

B. Increased software complexity

C. Increased automation

D. Elimination of maintenance costs

Hint

Classify the motive for the change before its implementation: repair, environment, useful capability or reduced future risk.

Answer B: Increased software complexity

Repeated changes without coherent boundaries can accumulate dependencies and make the evolving system harder to understand. Refactoring and good subsystem design help control that structural complexity while maintenance still requires impact analysis and regression checks. User engagement is a product-use outcome rather than the structural risk asked for. Automation is an execution technique, not itself structural deterioration, and ongoing evolution does not eliminate maintenance costs.

Why each choice is right or wrong

A - Incorrect: Engagement concerns user behavior rather than the internal structural deterioration asked for.

B - Correct: Accumulated poorly coordinated changes can increase structural complexity.

C - Incorrect: Automation does not itself describe deterioration of responsibilities or dependency structure.

D - Incorrect: Changing software still requires maintenance and regression work.

Primary note proof

Study-note section SE01

> Software engineering applies systematic methods to development, operation and maintenance. A software product includes executable code and relevant requirements, designs, test assets, configuration and user documentation. A process defines activities, ordering, roles, products and entry/exit conditions. Its four broad activities are specification, design/implementation, validation and evolution.

Study-note section SE40

Corrective maintenance fixes faults; adaptive maintenance responds to environment changes; perfective maintenance improves useful capability/performance; preventive maintenance reduces future risk, such as restructuring fragile code. Motive determines category; refactoring can support preventive or perfective work. Maintenance often lasts years and includes understanding legacy code and regression effort, so it may dominate lifecycle cost.

Study-note section SE22

Cohesion measures how well elements inside a module serve a related purpose; coupling measures dependency between modules. Seek high cohesion and low coupling. Shared mutable data and knowledge of another module's internal representation increase coupling. Information hiding exposes stable operations while concealing changeable decisions.

Study-note section SE01

poor object design can still be complex.

Study-note section SE43

Refactor to preserve behavior while improving structure; optimize only against measured needs; record rationale and update traceability and regression evidence after change.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 41 | lines 16-17

✓ Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous and the software must evolve to remain useful.

Software Engineering module | PDF page 19 | lines 24-29

5. Operation and maintenance: Normally (although not necessarily), this is the longest life cycle phase. Maintenance of a typical software product requires much more than the effort necessary to
develop the product itself. The system is installed and put into practical use. Maintenance involves
correcting errors which were not discovered in earlier stages of the life cycle, improving the implementation of system units and enhancing the system ‘s services as new requirements are discovered.

Software Engineering module | PDF page 120 | lines 6-12

6.1.3. Coupling and coherence
Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Question sources

Mock exam 4 · Question 51

Question 27 | Q00250

Which of the following best describes the difference between CPM and PERT?

A. CPM focuses on minimizing cost, while PERT focuses on minimizing time.

B. CPM is used for projects with uncertain activity durations, while PERT is used for projects with known activity durations.

C. PERT is used for projects with uncertain activity durations, while CPM is used for projects with known activity durations.

D. PERT is deterministic, whereas CPM is probabilistic.

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer C: PERT models uncertain durations; traditional CPM uses point durations.

SE30 gives traditional CPM point estimates and PERT optimistic, most-likely and pessimistic duration estimates. PERT therefore represents duration uncertainty, whereas the traditional CPM calculation uses known or fixed activity durations. The cost-versus-time contrast is not the defining distinction here, and the other two choices reverse the described treatment of uncertainty.

Why each choice is right or wrong

A - Incorrect: The note's defining contrast concerns duration uncertainty rather than assigning cost exclusively to CPM.

B - Incorrect: This reverses the models: PERT explicitly represents uncertain activity durations.

C - Correct: PERT uses uncertain three-point estimates, while traditional CPM uses fixed point durations.

D - Incorrect: The probabilistic/deterministic characterization is reversed relative to the note.

Primary note proof

Study-note section SE30

> Traditional CPM uses point durations; PERT models uncertainty with optimistic a, most-likely m and pessimistic b. The classic estimate is TE=(a+4m+b)/6 and variance=((b-a)/6)^2. For 3,6,15 days, TE=42/6=7 days and variance=4 days^2. Do not replace the weighted estimate by a simple average. These formulas are approximations, not guaranteed actual completion times.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 3-12

4.6. Scheduling, Organization and Team Structures
Project-task scheduling is an important project planning activity. It involves deciding which tasks would
be taken up when. In order to schedule the project activities, a software project manager needs to do
the following:
• Identify all the tasks needed to complete the project and Break down large tasks into small activities.
• Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
• Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Mock exam 4 · Question 52

Question 28 | Q00251

A key benefit of using a project management tool in software development is:

A. Automatic bug detection and fixing.

B. Replacing the need for human project managers.

C. Improved communication and collaboration within the team.

D. Increased coding efficiency for developers.

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer C: Improved team communication and collaboration.

SE28 says planning tools coordinate ownership, dependencies and progress while leaving judgment to people. That coordination supports communication and collaboration. The tool does not automatically detect and repair code faults, replace project managers or directly guarantee faster coding. Those benefits would require other capabilities and evidence beyond project-management tooling itself.

Why each choice is right or wrong

A - Incorrect: Project-management tools do not automatically detect and repair software defects.

B - Incorrect: They support planning and coordination while human managers still supply judgment.

C - Correct: Shared ownership, dependency and progress information helps the team communicate and collaborate.

D - Incorrect: A direct increase in coding efficiency is not the general coordination benefit established here.

Primary note proof

Study-note section SE28

Planning is continuous: identify scope/deliverables, decompose work, estimate effort/duration/cost, assign resources, sequence dependencies and revise as evidence changes. A WBS is hierarchical and covers project scope without adding unrelated work. Tools help coordinate ownership, dependencies and progress; they do not replace judgment or automatically fix bugs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 90 | lines 7-24

4.2. Project Planning
• Probably the most time-consuming project management activity.
• Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
• Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
The project plan sets out:
• The resources available to the project;
• The work breakdown;
• A schedule for the work.
Project Plan Structure
• Introduction.
• Project organisation.
• Risk analysis.
• Hardware and software resource requirements.
• Work breakdown.
• Project schedule.
• Monitoring and reporting mechanisms.

Software Engineering module | PDF page 81 | lines 11-21

3.4.2. Negotiating specifications with clients: Joint Application Design
Joint Application Design (JAD) is effectiveness lies in that the requirements elicitation work is done in one single workshop session in which all stakeholders participate. Users, clients, developers, and a trained session leader sit together in one room to present their viewpoint, listen
to other viewpoints, negotiate, and agree on a mutually acceptable solution. The outcome of the workshop, the final JAD document, is a complete system specification document that includes definitions of data elements, work flows, and interface screens. Because the final document is jointly developed by the stakeholders (that is, the participants who not only have an interest in the
success of the project, but also can make substantial decisions) the final JAD document represents

an agreement between users, clients, and developers, and thus minimizes requirements changes later in the development process.

Question sources

Mock exam 4 · Question 53

Question 29 | Q00252

A software development team is setting up a configuration management process. They need to identify the configuration items (CIs) for their project. Which of the following would not typically be considered a configuration item?

A. Internet browsers used by the development team

B. Source code files

C. User manuals

D. Test plans

Hint

Separate artifact history and controlled changes from scheduling, collaboration benefits and automated execution.

Answer A: Ordinary browsers used by the team are usually tools, not product configuration items.

Controlled configuration items include source files, requirements, manuals and test plans. A team's ordinary browser is normally just a tool. A particular browser version can become an environment CI if the project deliberately controls it for reproducible testing, but the word typically makes A the appropriate exception in the stated scenario.

Why each choice is right or wrong

A - Correct: An ordinary team browser is usually a tool; a deliberately controlled test-browser version would be an exception.

B - Incorrect: Source code files are standard controlled product configuration items.

C - Incorrect: User manuals can be versioned and baselined alongside the software product.

D - Incorrect: Test plans are controlled project artifacts whose versions must match the tested configuration.

Primary note proof

Study-note section SE33

Configuration management identifies controlled items, versions/baselines, change procedures, status and audits. Items may include source code, requirements, manuals, test plans, environments and build configurations. A team’s ordinary browser is usually a tool rather than product CI, but a controlled test-browser version may be an environment CI.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 142 | lines 5-10

> Configuration management avoids faults caused by undisciplined change in the system models. For example, it is a common mistake to a change a subsystem interface without notifying all developers of calling components. Configuration management can make sure that, if analysis models and code are becoming inconsistent with one another, analysts and implementor are notified. The assumption behind configuration management is that the system contains many less
> faults if change is controlled.

Software Engineering module | PDF page 89 | lines 17-24

> Assigning a role to a participant usually translates into assigning the responsibility of a specific work product. A work product is an artifact that is produced during the development, such as an internal document for other project participants or a deliverable to the customer.
> An analyst is responsible for a section of the RAD describing the functionality of the system. A system architect is responsible for the SDD describing the system's high-level architecture. A project manager is responsible for the SPMP describing the development processes. Finally, the system and its accompanying documentation also constitute a set of work products that are delivered to the client.

Question sources

Mock exam 4 · Question 54

Question 30 | Q00253

Which of the following is NOT a characteristic of a well-defined Work Breakdown Structure (WBS)?

A. It can be used to assign ownership and track the progress of project tasks.

B. It clearly defines deliverables for each level of the breakdown.

C. It includes activities that are outside the project scope.

D. It is organized in a hierarchical manner, with the overall project at the top and smaller tasks at the bottom.

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer C: It includes activities that are outside the project scope.

Planning breaks down the work needed to complete this project. Work outside project scope is therefore not part of its WBS; hierarchy and assignable tasks are consistent with decomposition.

Why each choice is right or wrong

A - Incorrect: Task assignment and tracking serve project management. See PDF p. 90, 95.

B - Incorrect: Deliverables constrain and clarify project work. See PDF p. 90, 95.

C - Correct: Outside-scope activities do not belong to the work needed to complete the defined project. See PDF p. 90, 95.

D - Incorrect: Breaking large tasks into smaller activities supports the hierarchy described. See PDF p. 90, 95.

Primary note proof

Study-note section SE28

Planning is continuous: identify scope/deliverables, decompose work, estimate effort/duration/cost, assign resources, sequence dependencies and revise as evidence changes. A WBS is hierarchical and covers project scope without adding unrelated work. Tools help coordinate ownership, dependencies and progress; they do not replace judgment or automatically fix bugs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 90 | lines 7-24

4.2. Project Planning
• Probably the most time-consuming project management activity.
• Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
• Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
The project plan sets out:
• The resources available to the project;
• The work breakdown;
• A schedule for the work.
Project Plan Structure
• Introduction.
• Project organisation.
• Risk analysis.
• Hardware and software resource requirements.
• Work breakdown.
• Project schedule.
• Monitoring and reporting mechanisms.

Software Engineering module | PDF page 95 | lines 3-12

4.6. Scheduling, Organization and Team Structures
Project-task scheduling is an important project planning activity. It involves deciding which tasks would
be taken up when. In order to schedule the project activities, a software project manager needs to do

the following:
• Identify all the tasks needed to complete the project and Break down large tasks into small activities.
• Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
• Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Mock exam 4 · Question 55

Question 31 | Q00254

A project team is working on a software project with multiple developers. They decided to implement a version control system (VCS). Which of the following scenarios best describes the use case for a distributed VCS like Git?

A. The team prefers to avoid merge conflicts by locking files so only one developer can work on a file at a time.

B. The team is large, geographically distributed, and each developer needs to work on features independently and merge changes later.

C. The team is small, and all developers work in the same physical location on a single branch.

D. The project requires strict control over who can make changes, and all changes must be approved by a central authority before being committed.

Hint

Separate artifact history and controlled changes from scheduling, collaboration benefits and automated execution.

Answer B: Independent distributed development followed by merging.

SE33 says a distributed Git clone contains local history and allows independent branches that can later be merged. B matches that working arrangement directly. Git does not require everyone to lock files, use one local branch or obtain central permission before every local commit. Additional review policies can be imposed, but they do not define distributed version control.

Why each choice is right or wrong

A - Incorrect: Exclusive file locking is not the defining distributed Git collaboration model.

B - Correct: Local histories and independent branches support geographically distributed work followed by merging.

C - Incorrect: A colocated team can use Git, but this scenario does not express its independent distributed-work benefit.

D - Incorrect: Central approval policies can exist, but they do not define or require distributed local commits.

Primary note proof

Study-note section SE33

> Change management handles request, impact assessment, decision/prioritization, implementation and validation; configuration management keeps all affected versions consistent. Distributed Git gives each clone local history and supports independent branches followed by merging; it does not eliminate merge conflicts or approvals. Plan test configuration management early and apply it throughout testing, not at just one exclusive phase.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 142 | lines 5-10

> Configuration management avoids faults caused by undisciplined change in the system models. For example, it is a common mistake to a change a subsystem interface without notifying all developers of calling components. Configuration management can make sure that, if analysis models and code are becoming inconsistent with one another, analysts and implementor are notified. The assumption behind configuration management is that the system contains many less
> faults if change is controlled.

Question sources

Mock exam 4 · Question 56

Question 32 | Q00255

The given statement describes which type of requirement? "A heat sensor detects an intrusion and alerts the system."

A. Known Requirement

B. None of the given options

C. Functional

D. Non-Functional

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Functional

Detection followed by an alert specifies an externally observable service, not a timing or quality constraint.

Why each choice is right or wrong

A - Incorrect: Known requirement is not the module's classification for this behavior. See PDF p. 65, 66.

B - Incorrect: A functional option is present. See PDF p. 65, 66.

C - Correct: Describes the system's required interaction with the intrusion sensor and alert recipient. See PDF p. 65, 66.

D - Incorrect: No performance or other quality bound is stated. See PDF p. 65, 66.

Primary note proof

Study-note section SE09

> A functional requirement states an observable service or behavior, including reactions to inputs, errors and forbidden actions. “When the sensor detects intrusion, send an alert” is functional. “Only a manager may approve refunds” also specifies permitted/prohibited behavior. A negative statement is not automatically nonfunctional.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 65 | lines 19-23

> 3.2.1. Functional Requirements
> Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. For example, the following is an example of functional requirements for SatWatch, a watch that resets itself without user intervention:

Software Engineering module | PDF page 66 | lines 14-18

> 3.2.2. Non-Functional and Pseudo-Requirements
> Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Nonfunctional requirements include quantitative constraints, such as response time (i.e., how fast the system reacts to user commands)
> or accuracy (i.e., how precise are the system’s numerical answers). The following are the

Question sources

Mock exam 4 · Question 57

Question 33 | Q00256

In the ordinary software lifecycle terminology used here, which document organizes the agreed user and detailed system requirements?

A. Software Design Description

B. Software Requirement Specification

C. Capability Development Document

D. Domain-Driven Design

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: Software Requirement Specification

The SRS records the agreed software services and constraints, including user-level and detailed system requirements. A Software Design Description explains how the solution is organized rather than being the requirements agreement. A Capability Development Document is a specialized defense capability-requirements artifact. Domain-driven design is an approach to organizing software models around business concepts. Thus the ordinary software requirements document requested is the SRS.

Why each choice is right or wrong

A - Incorrect: A design description records the solution structure rather than the agreed requirements specification.

B - Correct: The SRS organizes agreed user and system services and constraints.

C - Incorrect: A Capability Development Document belongs to a specialized defense capability-acquisition context.

D - Incorrect: Domain-driven design is a modeling/design approach, not the name of the requirements document.

Primary note proof

Study-note section SE08

> User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints. The Software Requirements Specification (SRS) organizes the agreed requirements.

Study-note section SE21

> System design turns the analysis model into architecture: design goals, subsystem responsibilities, dependencies, hardware mapping, data management, access control, control flow and startup/shutdown/error policies.

Study-note section SE-S06

> A Software Requirements Specification records agreed software services and constraints. A Capability Development Document is a defense capability-requirements artifact describing needed capability characteristics and performance parameters; its specialized acquisition context differs from the ordinary software SRS. Domain-driven design is an approach that organizes software models around the business domain, not the name of an agreed requirements specification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 26 | lines 10-17

3. Requirement’s specification: Requirements specification is the activity of translating the information gathered during the analysis activity in to a document that defines a set of requirements. Two types of requirements may be included in this document. User requirements are
abstract statements of the system requirements for the customer and end-user of the system; system
requirements are a more detailed description of the functionality to be provided.
4. Requirement’s validation: This activity checks the requirements for realism, consistency, and completeness. During this process, errors in the requirements document are inevitably discovered.
It must then be modified to correct these problems.

Software Engineering module | PDF page 18 | lines 13-20

System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
Software design involves identifying and describing the fundamental software system abstractions
and their relationships. The goal of the design phase is to transform the requirements specified in
the SRS document into a structure that is suitable for implementation in some programming language. In technical terms, during the design phase the software architecture is derived from the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Question sources

Mock exam 4 · Question 58

Question 34 | Q00257

Which listed statements describe supported advantages or uses of incremental delivery? Select all that apply. Interpret easier testing/debugging as a potential benefit of smaller changes, not a guarantee.

A. Easier to test and debug

B. Easier to test and debug & It is used when there is a need to get a product to the market early

C. Customer can respond to each increment

D. It is used when there is a need to get a product to the market early

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A, B, C, D: Easier to test and debug; Easier to test and debug & It is used when there is a need to get a product to the market early; Customer can respond to each increment; It is used when there is a need to get a product to the market early

Incremental delivery adds usable capability in stages. Smaller changes can be easier to test and debug, users can provide feedback after increments, and prioritized partial capability can reach the market early. Thus A, C and D each describe a supported benefit, and B is also true because it combines two true points. These benefits still require architecture, integration and testing effort.

Why each choice is right or wrong

A - Correct: Smaller increments can be easier to test and debug.

B - Correct: Both the smaller-change and early-delivery points are supported.

C - Correct: Users can respond to each usable increment.

D - Correct: Early delivery of prioritized capability is a supported use.

Primary note proof

Study-note section SE04

> Choose incremental delivery when prioritized partial capability must reach users early. Smaller changes can be easier to debug and permit feedback, though architecture and integration still require planning. Iteration need not rebuild the whole system every time, and increments need not be perfectly independent modules. Broad descriptions of early working software can fit prototypes, iterative or Agile processes; identify the purpose of that early version.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 22 | lines 24-29

> Advantages of Incremental Model:
> • Generates working software quickly and early during the software life cycle. This model is more flexible - less costly to change scope and requirements. It is easier to test and debug during a smaller iteration.
> • In this model customer can respond to each built. Lowers initial delivery cost.
> • Easier to manage risk because risky pieces are identified and handled during its iteration.

Question sources

Mock exam 4 · Question 59

Question 35 | Q00258

Use a process diagram whose named stages are requirements gathering, quick design, prototype construction, prototype refinement and engineering the production product. Which offered activity is not a separately named stage, although it can occur within prototype construction?

A. Coding

B. Prototype Refinement

C. Engineer Product

D. Quick Design

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A: Coding

Coding can be part of building an executable prototype even if the process diagram labels that work construction. It is therefore the offered activity that is not a separately named stage in the stated convention. Quick design, refinement and production engineering are named stages there. This answer concerns the diagram's labels and does not claim that prototyping excludes coding or always follows one universal phase taxonomy.

Why each choice is right or wrong

A - Correct: Coding can occur inside construction without a separate stage label.

B - Incorrect: Prototype refinement is a named stage.

C - Incorrect: Engineering the production product is a named stage.

D - Incorrect: Quick design is a named stage.

Primary note proof

Study-note section SE07

> A prototype explores requirements, usability or technical feasibility. A throwaway prototype is discarded after learning; an evolutionary prototype is deliberately refined toward production. A quick prototype may omit robustness, security or architecture, so do not deploy it unchanged without evaluation. Coding may occur during prototyping even if a diagram labels that activity construction rather than coding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 82 | lines 23-29

> • Prototype test. During this type of test, one or more users are presented with a piece of software that practically implements the system. A vertical prototype implements a complete use case through the system, and a horizontal prototype presents an interface for most use cases (without providing much or any functionality). The advantages of prototype

tests are that they provide a realistic view of the system to the user and that prototypes can be instrumented to collect detailed data. The disadvantages of prototypes are that they are expensive to build and to modify.

Question sources

Mock exam 4 · Question 60

Question 36 | Q00259

What is the major drawback of using the RAD Model?

A. Increases reusability of components, Highly specialized & skilled developers/designers are required

B. Increases reusability of components

C. Highly specialized & skilled developers/designers are required

D. Encourages customer/client feedback

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: The need for highly skilled developers and designers.

SE07 describes RAD as rapid construction using timeboxes, user participation and often reusable components, but requiring skilled designers and developers. That staffing prerequisite is the listed drawback. Reuse and customer feedback are useful features. A combines the genuine prerequisite with a benefit, so C states the drawback accurately without misclassifying reuse.

Why each choice is right or wrong

A - Incorrect: It bundles a genuine staffing drawback with component reuse, which is generally a benefit.

B - Incorrect: Component reuse helps rapid construction and is not the drawback asked for.

C - Correct: RAD depends on skilled designers and developers, which can constrain staffing and feasibility.

D - Incorrect: Customer feedback supports learning and fit rather than being the stated RAD disadvantage.

Primary note proof

Study-note section SE07

RAD uses rapid construction, timeboxing, user participation and often reusable components; it requires available users and skilled designers/developers. Reuse-oriented development searches components, evaluates fit, may adjust requirements, designs around interfaces and builds

missing parts. Product-line assets may include requirements, architecture, components and tests; all are potentially reusable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 24 | lines 20-29

Advantages:
1. Reduced cost and risks
2. Faster delivery of the software (Accelerated development)
Disadvantages:
1. Compromising requirements
2. Some control over the system evolution is lost as new versions of the reusable components are not under the control of the organization using them.
3. Increased maintenance costs
4. Not-invented-here syndrome
5. Lack of tool support

Question sources

Mock exam 4 · Question 61

Question 37 | Q00260

Which requirements are the foundation from which quality is measured?

A. None of the mentioned

B. Hardware

C. Software

D. Programmers

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer C: Software

Conformance is evaluated against software behavior specifications, making software requirements the foundation for the quality comparison here.

Why each choice is right or wrong

A - Incorrect: A software requirements option is present. See PDF p. 140, 26.

B - Incorrect: Hardware alone does not specify expected software behavior. See PDF p. 140, 26.

C - Correct: Defines the behavior against which reliability and failure are judged. See PDF p. 140, 26.

D - Incorrect: Programmer preferences are not the behavior specification. See PDF p. 140, 26.

Primary note proof

Study-note section SE34

Quality assurance establishes and checks processes/standards; quality control evaluates work products. An SQA plan identifies standards, responsibilities, evaluations, reviews, audits and assurance records. Quality of conformance means following the design/specification; a product can conform to a poor specification. Verification asks whether specified obligations are implemented correctly; validation asks whether the product meets intended stakeholder needs. Both can use reviews, analysis or execution, so “human-only versus computer-only” is false.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 140 | lines 11-18

Reliability is a measure of success with which the observed behavior of a system conforms to the specification of its behavior. Software reliability is the probability that a software system will not cause the failure of the system for a specified time under specified conditions. Failure is any deviation of the observed behavior from the specified behavior. An error means the system is in a state such that further processing by the system will lead to a failure, which causes the system to
deviate from its intended behavior. A fault, also called defect or bug, is the mechanical or algorithmic cause of an error. The goal of testing is to maximize the number of discovered faults, which then allows developers to correct them and increase the reliability of the system.

Software Engineering module | PDF page 26 | lines 10-17

3. Requirement’s specification: Requirements specification is the activity of translating the information gathered during the analysis activity in to a document that defines a set of requirements. Two types of requirements may be included in this document. User requirements are
abstract statements of the system requirements for the customer and end-user of the system; system
requirements are a more detailed description of the functionality to be provided.
4. Requirement’s validation: This activity checks the requirements for realism, consistency, and completeness. During this process, errors in the requirements document are inevitably discovered.
It must then be modified to correct these problems.

Question sources

Mock exam 4 · Question 62

Question 38 | Q00261

Which of the following is not a SQA plan for a project?

A. Documents to be produced by the SQA group

B. Evaluations to be performed

C. Audits and reviews to be performed

D. Amount of technical work

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer D: Amount of general technical development work.

SE34 identifies assurance responsibilities, evaluations, reviews, audits and assurance records as SQA-plan contents. The documents produced by SQA are assurance records; evaluations, audits and reviews are assurance activities. The amount of general technical development work belongs primarily to project scope and effort planning rather than defining the SQA activities listed here.

Why each choice is right or wrong

A - Incorrect: Assurance documents and records are legitimate SQA-plan deliverables.

B - Incorrect: Planned evaluations are explicitly part of an SQA plan.

C - Incorrect: Reviews and audits are explicitly listed assurance-plan activities.

D - Correct: General technical-work quantity is a project scope/effort concern rather than the listed assurance-plan content.

Primary note proof

Study-note section SE34

> Quality assurance establishes and checks processes/standards; quality control evaluates work products. An SQA plan identifies standards, responsibilities, evaluations, reviews, audits and assurance records. Quality of conformance means following the design/specification; a product can conform to a poor specification. Verification asks whether specified obligations are implemented correctly; validation asks whether the product meets intended stakeholder needs. Both can use reviews, analysis or execution, so “human-only versus computer-only” is false.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 99 | lines 14-19

> 4.8. Quality Assurance Monitoring Plans
> Quality assurance consists of the auditing and reporting functions of management. The goal of quality assurance is to provide management with the data necessary to be informed about product
> quality, thereby gaining insight and confidence that product quality is meeting its goals. Of

course,
if the data provided through quality assurance identify problems, it is management's responsibility
to address the problems and apply the necessary resources to resolve quality issues.

Question sources

Mock exam 4 · Question 63

Question 39 | Q00262

Which of the below is not a part of the Test Plan?

A. Schedule

B. Incident reports

C. Entry and exit criteria

D. Risk

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: Incident reports are execution outputs, rather than the test plan itself.

SE39 places objectives, scope, resources, schedule, risks and entry/exit criteria in test planning. Incident reports document observed problems during execution and are normally outputs of that work. They may be referenced by later plans, but that does not make actual incident reports an intrinsic part of the prospective test plan.

Why each choice is right or wrong

A - Incorrect: The test schedule is planned work and belongs in a test plan.

B - Correct: Incident reports normally record execution findings rather than describing the planned testing work.

C - Incorrect: Entry and exit criteria define when testing can start or finish and belong in planning.

D - Incorrect: Testing risks influence scope and priorities and are legitimate test-plan content.

Primary note proof

Study-note section SE39

A test plan describes intended work; incident/defect reports are normally execution outputs. A defect report should identify build/environment, steps, expected versus actual behavior, evidence and severity. Anyone finding a fault should record it; testers often coordinate reporting, but it is not their exclusive permission. Configuration control makes the tested version reproducible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 90 | lines 7-24

4.2. Project Planning
• Probably the most time-consuming project management activity.
• Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
• Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
The project plan sets out:
• The resources available to the project;
• The work breakdown;
• A schedule for the work.
Project Plan Structure
• Introduction.
• Project organisation.
• Risk analysis.
• Hardware and software resource requirements.
• Work breakdown.
• Project schedule.
• Monitoring and reporting mechanisms.

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Software Engineering module | PDF page 19 | lines 20-23

System testing is normally carried out in a planned manner according to the system test plan document. The system test plan identifies all testing-related activities that must be performed, specifies the schedule of testing, and allocates resources. It also lists all the test cases and the expected outputs for each test case.

Question sources

Mock exam 4 · Question 64

Question 40 | Q00263

Exploratory testing is a -----?

A. Experience-based Test Design Technique

B. Black Box Test Design Technique

C. White Box Test Design Technique

D. Grey Box Test Design Technique

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer A: Experience-based testing.

SE36 explicitly classifies exploratory testing as experience based because learning, test design and execution occur together. It may use knowledge of specifications or internals, but black-box, white-box and grey-box describe different bases of test derivation. None of them replaces the defining experience-based classification asked for in this question.

Why each choice is right or wrong

A - Correct: Exploratory testing uses experience while learning, designing and executing tests together.

B - Incorrect: It may use external behavior, but black-box alone is not its defining classification.

C - Incorrect: Exploratory testing does not require test cases to be derived from internal code structure.

D - Incorrect: Partial internal knowledge can help exploration, but it does not define the technique.

Primary note proof

Study-note section SE36

> Black-box techniques derive cases from specified behavior; white-box techniques derive them from internal structure; grey-box uses partial internal knowledge. Regression testing checks for unintended effects after change; confirmation testing checks the particular defect repair. Exploratory testing combines learning, design and execution and is experience based. Manual/automated describe execution support, not a compulsory black-/white-box classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 149 | lines 6-10

Equivalence testing is a black box testing technique that minimizes the number of test cases. The possible inputs are partitioned into equivalence classes, and a test case is selected for each class. The assumption of equivalence testing is that systems usually behave in similar ways for all members of a class. To test the behavior associated with an equivalence class, we only need to test
one member of the class. Equivalence testing consists of two steps: identification of the

Software Engineering module | PDF page 149 | lines 24-29

Path testing
Path testing is a white box testing technique that identifies faults in the implementation of the component. The assumption behind path testing is that, by exercising all possible paths through the code at least once, most faults will trigger failures. The identification of paths requires knowledge of the source code and data structures.
The starting point for path testing is the flow graph. A flow graph consists of nodes representing

Question sources

Mock exam 4 · Question 65

Question 41 | Q00264

Non-conformance to software requirements is known as

A. Software reliability

B. Software availability

C. None of the mentioned

D. Software failure

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer D: Software failure

Failure is the deviation of observed behavior from its specified behavior; reliability measures conformance rather than the non-conformance event.

Why each choice is right or wrong

A - Incorrect: Probability or measure of success, not the deviation itself. See PDF p. 140.

B - Incorrect: Availability is not defined as non-conformance. See PDF p. 140.

C - Incorrect: The failure option supplies the defined term. See PDF p. 140.

D - Correct: Matches deviation from specified behavior. See PDF p. 140.

Primary note proof

Study-note section SE34

A fault/defect is an underlying problem; an error may be an incorrect internal state; a failure is an externally observed departure from required service. Reliability is a product quality attribute; low cost or short development time are project outcomes. Tests increase confidence and reveal defects but cannot generally guarantee correctness over every possible input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 140 | lines 11-18

Reliability is a measure of success with which the observed behavior of a system conforms to the specification of its behavior. Software reliability is the probability that a software system will not cause the failure of the system for a specified time under specified conditions. Failure is any deviation of the observed behavior from the specified behavior. An error means the system is in a state such that further processing by the system will lead to a failure, which causes the system to
deviate from its intended behavior. A fault, also called defect or bug, is the mechanical or algorithmic cause of an error. The goal of testing is to maximize the number of discovered faults, which then allows developers to correct them and increase the reliability of the system.

Question sources

Mock exam 4 · Question 66

Question 42 | Q00308

When comparing internal subsystem-grouping criteria with an external user perspective, which option names the external perspective rather than an internal structural or responsibility criterion?

A. Coupling

B. System's functionality

C. Cohesion

D. Users thinking

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: Users thinking

Coupling measures dependencies across subsystem boundaries, cohesion measures relatedness within a subsystem, and functionality helps assign responsibilities. These are internal structural grouping criteria. A user’s perspective describes externally needed tasks and behavior, so it is the different kind of information requested. User workflows can still inform architecture; the distinction does not claim users have no influence on design or that internal boundaries should ignore their needs.

Why each choice is right or wrong

A - Incorrect: Coupling directly measures inter-subsystem dependency.

B - Incorrect: Functionality guides internal responsibility grouping.

C - Incorrect: Cohesion measures how well elements grouped inside a subsystem serve a related purpose.

D - Correct: This names the external user perspective rather than an internal structural measure or responsibility criterion.

Primary note proof

Study-note section SE21

> Group closely related objects and functionality, minimize associations crossing boundaries, and isolate communication infrastructure where useful. User workflows inform decomposition but do not alone dictate the internal structure. Analysis describes domain needs; subsystem decomposition is principally a design result. Architecture documentation supports communication and records major decisions, not every source-level execution step.

Study-note section SE22

> Cohesion measures how well elements inside a module serve a related purpose; coupling measures dependency between modules. Seek high cohesion and low coupling. Shared mutable data and knowledge of another module's internal representation increase coupling. Information hiding exposes stable operations while concealing changeable decisions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 116 | lines 16-29

> The analysis model describes the system completely from the actors' point of view and serves as the basis of communication between the client and the developers. The analysis model, however, does not contain information about the internal structure of the system, its hardware configuration,
> or, more generally, how the system should be realized. System design is the first step in this direction. System design results in the following products:
> • List of design goals, describing the qualities of the system that developers should optimize
> • Software architecture, describing the subsystem decomposition in terms of subsystem responsibilities, dependencies among subsystems, subsystem mapping to hardware, and major policy decisions such as control flow, access control, and data storage.
> The design goals are derived from the nonfunctional requirements. Design goals guide the decisions to be made by the developers especially when trade-offs are needed. The subsystem decomposition constitutes the bulk of system design. Developers divide the system into manageable pieces to deal with complexity: Each subsystem is assigned to a team and realized independently. In order for this to be possible, though, developers need to address system-wide

Software Engineering module | PDF page 120 | lines 6-12

> 6.1.3. Coupling and coherence
> Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely

coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Software Engineering module | PDF page 120 | lines 24-27

Coherence is the strength of dependencies within a subsystem. If a subsystem contains many objects that are related to each other and perform similar tasks, its coherence is high. If a subsystem
contains a number of unrelated objects, its coherence is low. A desirable property of a subsystem decomposition is that it leads to subsystems with high coherence.

Question sources

Computer Science model exam · Question 39

University of Gondar model exam · Question 39

Question 43 | Q00309

The functional model is represented graphically with --

A. Use case diagram

B. State transition diagram

C. Class diagram

D. Object diagram

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: Use case diagram

Use case diagrams represent the functional model; different diagrams represent the dynamic and object models.

Why each choice is right or wrong

A - Correct: Explicit graphical form of the functional model. See PDF p. 101.

B - Incorrect: Models dynamic state transitions. See PDF p. 101.

C - Incorrect: Models object structure. See PDF p. 101.

D - Incorrect: Models object instances, not the functional model. See PDF p. 101.

Primary note proof

Study-note section SE13

> A scenario is one concrete execution; a use case abstracts the related successful and alternative scenarios. Analysis integrates functional (use cases/scenarios), structural/object (classes) and dynamic (states/interactions) models. Entity objects represent domain information such as Incident; boundary objects mediate interaction, such as ReportForm; control objects coordinate a use case. A UI form is normally boundary, not entity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 101 | lines 5-6

> In UML, the functional model is represented with use case diagrams, the object model with class diagrams, and the dynamic model with statechart and sequence diagrams.

Question sources

Computer Science model exam · Question 40

University of Gondar model exam · Question 40

Question 44 | Q00310

Under the course’s distinction between problem analysis and solution design, which statement is false?

A. System Analysis Focuses on the user's view of the system

B. Design focuses on the developer point of view.

C. Result of system analysis is a model that includes a subsystem decomposition

D. Requirements elicitation establishes externally visible stakeholder needs and constraints before internal solution design.

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Result of system analysis is a model that includes a subsystem decomposition

Analysis clarifies the domain and externally required behavior; system design chooses the solution architecture and subsystem responsibilities. Attributing subsystem decomposition to the principal analysis result therefore confuses these stages. Requirements elicitation establishes stakeholder needs and constraints before the internal solution is selected. The revised elicitation option avoids the stronger claim that only one user’s viewpoint matters, while retaining the intended external-needs versus internal-design distinction.

Why each choice is right or wrong

A - Incorrect: Analysis clarifies the user/domain view of required behavior.

B - Incorrect: Design develops the solution structure and implementation-facing decisions.

C - Correct: Subsystem decomposition is principally a system-design result, so attributing it to analysis is the false claim.

D - Incorrect: Elicitation establishes externally visible stakeholder needs and constraints before choosing internal design.

Primary note proof

Study-note section SE02

> Analysis clarifies the problem; system design chooses architecture; object/detail design specifies interfaces and algorithms; implementation writes code; testing evaluates behavior; deployment puts it into service.

Study-note section SE21

> User workflows inform decomposition but do not alone dictate the internal structure. Analysis describes domain needs; subsystem decomposition is principally a design result.

Study-note section SE08

> User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 63 | lines 3-8

> Requirements elicitation and analysis focus only on the user's view of the system. For example, the system functionality, the interaction between the user and the system, the errors that the system
> can detect and handle, and the environmental conditions in which the system functions, are part of
> the requirements. The system structure, the implementation technology selected to build the system, the system design, the development methodology, and other aspects not directly visible to
> the user are not part of the requirements.

Software Engineering module | PDF page 116 | lines 16-29

> The analysis model describes the system completely from the actors' point of view and serves as the basis of communication between the client and the developers. The analysis model, however, does not contain information about the internal structure of the system, its hardware configuration,
> or, more generally, how the system should be realized. System design is the first step in this direction. System design results in the following products:
> • List of design goals, describing the qualities of the system that developers should optimize
> • Software architecture, describing the subsystem decomposition in terms of subsystem responsibilities, dependencies among subsystems, subsystem mapping to hardware, and major policy decisions such as control flow, access control, and data storage.
> The design goals are derived from the nonfunctional requirements. Design goals guide the decisions to be made by the developers especially when trade-offs are needed. The subsystem decomposition constitutes the bulk of system design. Developers divide the system into manageable pieces to deal with complexity: Each subsystem is assigned to a team and realized independently. In order for this to be possible, though, developers need to address system-wide

Question sources

Computer Science model exam · Question 41

University of Gondar model exam · Question 41

Question 45 | Q00311

Which of the following heuristics is not used to identify Entity objects

A. Real-world entities

B. Real-world activities

C. Forms

D. Data sources

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Forms normally identify boundary objects.

SE13 defines entity objects as domain information and gives ReportForm as a boundary mediating interaction. Thus forms normally suggest boundary objects rather than entity objects. Real-world things, domain activities represented as information and relevant stored data can suggest entity candidates; those are heuristics, not mechanical rules guaranteeing a class for every noun or activity.

Why each choice is right or wrong

A - Incorrect: Real-world domain entities are natural candidates for entity objects.

B - Incorrect: A domain activity represented as persistent domain information can suggest an entity candidate.

C - Correct: A user-interface form normally mediates interaction and is modeled as a boundary object.

D - Incorrect: Relevant domain data sources can suggest information-holding entities, subject to analysis rather than automatic conversion.

Primary note proof

Study-note section SE13

> Use nouns as candidate objects, possessive descriptions as candidate attributes, and verbs such as manages/reports-to as candidate associations. These are heuristics, not automatic parsing rules. CRC means Class, Responsibilities, Collaborators. A control per use case or participating actor can be a useful starting point, not a mandatory UML rule or fixed object lifetime.

Study-note section SE13

> A UI form is normally boundary, not entity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 105 | lines 9-15

> 5.2.1. Identifying Entity Objects
> Requirements Elicitation, participating objects are found by examining each use case and identifying candidate objects. Natural language analysis is an intuitive set of heuristics for identifying objects, attributes, and associations from a system specification. Abbott’s heuristics maps parts of speech (e.g., nouns, having verbs, being verbs, adjectives) to model components (e.g., objects, operations, inheritance relationships, classes). Table 4-1 provides examples of such mappings by examining the Report Emergency use case.

Software Engineering module | PDF page 101 | lines 9-15

The analysis object model consists of entity, boundary, and control objects. Entity objects represent the persistent information tracked by the system. Boundary objects represent the interactions between the actors and the system. Control objects represent the tasks that a formed by the user and supported by the system. In the 2B watch example, Year, Month, Day are entity objects; Button Boundary and LCD Display Boundary are boundary objects; Change Date Control is
a control object that represents the activity of changing the date by pressing combinations of buttons.

Question sources

Computer Science model exam · Question 42

University of Gondar model exam · Question 42

Question 46 | Q00312

"Consider a system where, a heat sensor detects an intrusion and alerts the security company." What kind of a requirement the system is providing?

A. Functional Requirement

B. Non-Functional

C. Constraints

D. Known Requirement

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: Functional Requirement

Detecting an intrusion and notifying the security company is a required interaction/service independent of implementation.

Why each choice is right or wrong

A - Correct: Specifies the system's required behavior. See PDF p. 65.

B - Incorrect: No response-time, reliability or quality limit is stated. See PDF p. 65.

C - Incorrect: No implementation or resource restriction is stated. See PDF p. 65.

D - Incorrect: Known Requirement is not the behavioral category defined here. See PDF p. 65.

Primary note proof

Study-note section SE09

Write one clear obligation with an actor/trigger, action and observable result. “The system shall generate a PDF report” is functional. “The system shall not reveal another customer’s balance” can be a security-related functional rule. Security as a broad quality goal and the functions

implementing it must be distinguished. Words such as must, should and not work do not by themselves determine a requirement’s classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 65 | lines 19-23

3.2.1. Functional Requirements
Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. For example, the following is an example of functional requirements for SatWatch, a watch that resets itself without user intervention:

Question sources

Computer Science model exam · Question 43

University of Gondar model exam · Question 43

Question 47 | Q00313

which one of the following is used to Identifying Actors

A. What kind of dependencies are exist among use cases.

B. What information does the actor access?

C. Which user groups execute the system's main functions?

D. What are the tasks that the actor wants the system to perform?

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Which user groups execute the system's main functions?

Identifying user groups that execute the main functions identifies actors; questions about information and tasks identify scenarios.

Why each choice is right or wrong

A - Incorrect: Concerns use-case relationships rather than identifying external entities. See PDF p. 71, 73.

B - Incorrect: Listed as a scenario-identification question. See PDF p. 71, 73.

C - Correct: Exact actor-identification heuristic. See PDF p. 71, 73.

D - Incorrect: Listed as a scenario-identification question. See PDF p. 71, 73.

Primary note proof

Study-note section SE14

An actor is an external role interacting with the system; it can be a person, device or another system. A person may play several roles, and several people may play one role. A primary actor seeks a goal; a supporting actor provides a service. Ask which user groups perform main functions, administer or maintain the system, and which external systems exchange information.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 71 | lines 13-18

When identifying actors, developers can ask the following questions:
Questions for identifying actors
• Which user groups are supported by the system to perform their work?
• Which user groups execute the system's main functions?
• Which user groups perform secondary functions, such as maintenance and administration?
• Will the system interact with any external hardware or software system?

Software Engineering module | PDF page 73 | lines 4-9

Questions for identifying scenarios
• What are the tasks that the actor wants the system to perform?
• What information does the actor access? Who creates that data? Can it be modified or removed?
By whom?
• Which external changes does the actor need to inform the system about? How often? When?
• Which events does the actor need to be informed by the system about? With what latency?

Question sources

Computer Science model exam · Question 44

University of Gondar model exam · Question 44

Question 48 | Q00547

_____is requirement elicitation techniques that is used to gather the requirements using day-to-day working observation and notes made of the actual tasks.

A. Use case

B. Ethnography

C. Scenario

D. Interview

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: Ethnography.

The stem describes observing people's actual daily tasks and recording what they do. SE12 names that observation-based elicitation approach ethnography. Interviews ask stakeholders questions, while scenarios and use cases describe executions or goals; those can complement observation but do not name the day-to-day fieldwork technique itself.

Why each choice is right or wrong

A - Incorrect: A use case describes goal-oriented interactions rather than naming observational fieldwork.

B - Correct: Ethnography gathers understanding by observing actual day-to-day work and recording it.

C - Incorrect: A scenario is one concrete execution, not the observation-based elicitation technique itself.

D - Incorrect: An interview asks people questions rather than primarily observing their daily tasks.

Primary note proof

Study-note section SE12

> Interviews ask stakeholders about needs; document analysis examines manuals, contracts and current records; observation/ethnography studies actual daily work. Prototypes reveal tacit expectations. Workshops/JAD bring users, managers and developers together with a facilitator and recorder to build consensus. Use several sources and check assumptions against observed work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 80 | lines 5-11

> Task analysis is based on the assumption that it is inefficient to ask users to describe what they do
> and how they do it. Users usually do not think explicitly about the sequence of tasks that are required to accomplish their work as they have often repeated these tasks many times. Users, when
> ask how they accomplish their work, would describe, at best, how they are supposed to accomplish
> it, which may be far from reality. Consequently, task analysis uses observation as an alternative to
> build an initial task model. This initial task model is then refined by asking the users why they accomplish a task a certain way.

Question sources

Computer Science model exam · 2015 · Question 16

Question 49 | Q00549

Which one of the following is more related to effective management of software complexity?

A. Modularity

B. Reusability

C. Extensibility

D. All

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer A: Modularity

Decomposition into simpler, manageable subsystems controls complexity. Reuse and extensibility can benefit designs, but do not substitute for this direct mechanism.

Why each choice is right or wrong

A - Correct: Divides the solution domain into manageable simpler parts. See PDF p. 118, 32.

B - Incorrect: Reusability is an additional benefit of object-oriented modules. See PDF p. 118, 32.

C - Incorrect: Extensibility concerns future additions rather than directly partitioning complexity. See PDF p. 118, 32.

D - Incorrect: The text directly establishes modularity; it does not identify all three as equivalent complexity mechanisms. See PDF p. 118, 32.

Primary note proof

Study-note section SE21

> System design turns the analysis model into architecture: design goals, subsystem responsibilities, dependencies, hardware mapping, data management, access control, control flow and startup/shutdown/error policies. Detail/object design refines APIs, parameter/return types, exceptions, component choices and algorithms. A subsystem is a manageable logical collection with services and an interface; it need not be a physical replaceable artifact.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 118 | lines 11-16

> 6.1.1. Subsystems and classes
> In order to reduce the complexity of the application domain, we identified smaller parts called classes and organized them into packages. Similarly, to reduce the complexity of the solution domain, we decompose a system into simpler parts, called subsystems, which are made of a number of solution domain classes. In the case of complex subsystems, we recursively apply this principle and decompose a subsystem into simpler subsystems (see Figure 6.2).

Software Engineering module | PDF page 32 | lines 12-21

A class in an object-oriented system carefully delineates between its interfaces the routines and attributes within a class are held together tightly. In a properly designed system, the classes will be grouped into subsystems but remain independent; therefore, changing one class has no impact
on other classes, and so, the impact is minimized. However, the object-oriented approach is not a panacea; nothing is magical here that will promote perfect design or perfect code.
Promotion of reusability
Objects are reusable because they are modeled directly out of a real-world problem domain. Each
object stands by itself or within a small circle of peers (other objects). Within this framework, the class does not concern itself with the rest of the system or how it is going to be used within a particular system.

Question sources

Computer Science model exam · 2015 · Question 18

Question 50 | Q00551

Which one of the following testing required knowledge about internal structure, design and implementation of the system?

A. Black box Testing

B. White box testing

C. Alpha testing

D. Beta testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: White box testing

Path testing requires source code and data-structure knowledge; black-box equivalence testing instead groups observable inputs.

Why each choice is right or wrong

A - Incorrect: Does not require internal paths. See PDF p. 149, 149.

B - Correct: Matches the source-knowledge requirement. See PDF p. 149, 149.

C - Incorrect: Alpha denotes development-team system testing, not an internal-knowledge technique. See PDF p. 149, 149.

D - Incorrect: Beta denotes testing by customers, not an internal-knowledge technique. See PDF p. 149, 149.

Primary note proof

Study-note section SE36

Black-box techniques derive cases from specified behavior; white-box techniques derive them from internal structure; grey-box uses partial internal knowledge. Regression testing checks for unintended effects after change; confirmation testing checks the particular defect repair. Exploratory testing combines learning, design and execution and is experience based. Manual/automated describe execution support, not a compulsory black-/white-box classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 149 | lines 24-29

Path testing
Path testing is a white box testing technique that identifies faults in the implementation of the component. The assumption behind path testing is that, by exercising all possible paths through the code at least once, most faults will trigger failures. The identification of paths requires knowledge of the source code and data structures.
The starting point for path testing is the flow graph. A flow graph consists of nodes representing

Software Engineering module | PDF page 149 | lines 6-10

Equivalence testing is a black box testing technique that minimizes the number of test cases. The possible inputs are partitioned into equivalence classes, and a test case is selected for each class. The assumption of equivalence testing is that systems usually behave in similar ways for all members of a class. To test the behavior associated with an equivalence class, we only need to test
one member of the class. Equivalence testing consists of two steps: identification of the

Question sources

Computer Science model exam · 2015 · Question 20

Question 51 | Q00554

Which team organization shares development decisions among team members rather than centralizing technical coordination in a chief programmer?

A. Functional

B. Project

C. Democratic

D. Chief programmer

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer C: Democratic

Democratic teams share decisions, whereas chief-programmer teams centralize technical coordination. Functional and project organizations classify how people are grouped, not how authority is distributed. Asking about shared decision-making therefore makes democratic the unique answer. It does not require every individual to perform every lifecycle task; a democratic team can still divide work by expertise and responsibilities.

Why each choice is right or wrong

A - Incorrect: Functional organization groups specialists by discipline.

B - Incorrect: Project organization groups people by project rather than defining shared authority.

C - Correct: Democratic teams share decisions.

D - Incorrect: Chief-programmer teams centralize technical coordination.

Primary note proof

Study-note section SE32

> Functional organizations group specialists by discipline; project organizations group people around projects. Functional specialization helps staffing and knowledge continuity but may create handoff barriers. Democratic teams share decisions; chief-programmer teams centralize technical coordination. These are different classification dimensions. Tuckman’s sequence is forming→storming→norming→performing→adjourning: orientation, conflict, norms, effective work and closure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 20-26

> Functional format vs. project format
> • There are essentially two broad ways in which a software development organization can be structured:
> • Functional format and project format. In the project format, the project development staffs are divided based on the project for which they work (as shown in next fig).
> • In the functional format, the development staffs are divided based on the functional group to which they belong.

Software Engineering module | PDF page 96 | lines 3-11

> Team Structure
> • Team structure addresses the issue of organization of the individual project teams. There are some possible ways in which the individual project teams can be organized.
> • Problems of different complexities and sizes often require different team structures for chief solution.
> • The most common types of team structure are:
> · Chief Programmer Team
> · Democratic Team
> · Mixed Control Team Organization

Question sources

Computer Science model exam · 2015 · Question 23

Question 52 | Q00555

Which capability maturity model level an organization must master the requirements management software quality assurance, and software project planning

A. Repeatable level

B. Initial level

C. Defined level

D. Managed level

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer A: Historical CMM Level 2: Repeatable.

SE32 explicitly assigns requirements management, software project planning, quality assurance and configuration management to historical Software CMM Level 2. Its name is Repeatable, rather than the later CMMI label Managed. The other named choices correspond to levels 1, 3 and 4 in this historical model, so A matches both practices and terminology.

Why each choice is right or wrong

A - Correct: Repeatable is historical Software CMM Level 2, which establishes the listed basic project practices.

B - Incorrect: Initial is historical Level 1, before the repeatable project discipline specified here.

C - Incorrect: Defined is historical Level 3, not the level associated with this basic practice list.

D - Incorrect: Managed is historical Software CMM Level 4; modern CMMI terminology must not be substituted.

Primary note proof

Study-note section SE32

> Historical Software CMM levels are 1 Initial, 2 Repeatable, 3 Defined, 4 Managed, 5 Optimizing. Level2 establishes basic project discipline including requirements management, project planning, quality assurance and configuration management. These historical labels must not be silently replaced by modern CMMI terminology. A maturity model concerns process capability, not a guarantee that every product is defect free.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 91 | lines 37-42

• Software Metrics can be used to indicate the basic attributes of the software. The indicative metrics, not exhaustive, are
• Size - Line of code ,Function point count
• Quality- Reliability, Accuracy, Dependability, and these are measured through some measures like errors, failures and number of runs between failures ,etc.
• Execution Time -process time of execution of a critical process and many more.

Question sources

Computer Science model exam · 2015 · Question 24

Question 53 | Q00558

All are the advantages of functional format structure, except

A. Ease of staffing

B. Production of good quality documents

C. No need job specialization

D. Efficient handling manpower turnover

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer C: No need for job specialization is false of a functional organization.

SE32 defines functional organizations as grouping specialists by discipline. Claiming that specialization is unnecessary contradicts that organizing principle. Ease of staffing and knowledge continuity are benefits of specialization; continuity can help with turnover. Specialist expertise can also support documentation quality, although no organizational form guarantees good documents. C is the clear exception.

Why each choice is right or wrong

A - Incorrect: Grouping specialists supports staffing within established disciplines, so this is a plausible benefit.

B - Incorrect: Specialist expertise can support good documentation; this is not the direct contradiction of functional grouping.

C - Correct: This contradicts the defining use of specialist disciplines in a functional organization.

D - Incorrect: Knowledge continuity within a functional group can help absorb staff turnover.

Primary note proof

Study-note section SE32

Functional organizations group specialists by discipline; project organizations group people around projects. Functional specialization helps staffing and knowledge continuity but may create handoff barriers. Democratic teams share decisions; chief-programmer teams centralize

technical coordination. These are different classification dimensions. Tuckman's sequence is forming→storming→norming→performing→adjourning: orientation, conflict, norms, effective work and closure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 20-26

Functional format vs. project format
• There are essentially two broad ways in which a software development organization can be structured:
• Functional format and project format. In the project format, the project development staffs are divided based on the project for which they work (as shown in next fig).
• In the functional format, the development staffs are divided based on the functional group to which they belong.

Question sources

Computer Science model exam · 2015 · Question 27

Question 54 | Q00568

Which testing should focus on testing the functionality of objects or methods?

A. Integration testing

B. White box testing

C. Black box testing

D. Unit testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer D: Unit testing

Unit testing isolates the building blocks, including objects; a method is part of such an individual component's behavior.

Why each choice is right or wrong

A - Incorrect: Tests components together. See PDF p. 148, 144.

B - Incorrect: A technique based on internal paths, not the requested component level. See PDF p. 148, 144.

C - Incorrect: A technique based on external behavior, not a component level. See PDF p. 148, 144.

D - Correct: Tests isolated objects or subsystems and their functionality. See PDF p. 148, 144.

Primary note proof

Study-note section SE35

Acceptance testing supports the customer/user's acceptance decision against agreed needs and criteria. Alpha typically happens in a controlled developer setting with selected users; beta uses external users in their environments. None of these levels is universally the longest. Unit is a level, while white-box is a technique; a unit can be tested using black-box cases too.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 148 | lines 19-26

Unit testing focuses on the building blocks of the software system, that is, objects and subsystems.
There are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall test activities, allowing us to focus on smaller units of the system. Second, unit testing makes it easier to pinpoint and correct faults, given that few components are involved in the test. Third, unit testing allows parallelism in the testing activities; that is, each component can be tested independently of one another.
The specific candidates for unit testing are chosen from the object model and the system decomposition of the system. In principle, all the objects developed during the development

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Question sources

Computer Science model exam · 2015 · Question 37

Question 55 | Q00572

Which of the following views represents the interaction of the user with the software but tells nothing about the internal working of the software?

A. Use case diagram

B. Activity diagram

C. Class diagram

D. Sequence diagram

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer A: Use case diagram

A use case describes external behavior from an actor's point of view without prescribing the internal realization.

Why each choice is right or wrong

A - Correct: Directly models actor-visible interaction. See PDF p. 47, 54, 51.

B - Incorrect: Models action/control flow and may include internal work. See PDF p. 47, 54, 51.

C - Incorrect: Describes classes and their structure. See PDF p. 47, 54, 51.

D - Incorrect: Distributes interaction among participating objects, revealing internal collaboration. See PDF p. 47, 54, 51.

Primary note proof

Study-note section SE14

> Draw actors outside the system boundary rectangle and use-case ovals inside. Name a goal with a verb and noun. The diagram shows roles, goals and relationships, not the full detailed algorithm or UI layout. Being an internal component is different from being an external actor; the chosen boundary determines which role applies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 47 | lines 30-39

> Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
> Use cases describe the behavior of the system as seen from an actor's point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate a
> use case to access the system functionality. The use case can then initiate other use cases and

gather
more information from the actors. When actors and use cases exchange information, they are said
to communicate. We will see later that we represent these exchanges with communication relationships.

Software Engineering module | PDF page 54 | lines 26-35

both buttons simultaneously. Figure 2.10 depicts a sequence diagram for an actor setting his 2Bwatch one minute ahead.
Each column represents an object that participates in the interaction. The vertical axis represents
time from top to bottom. Messages are shown by arrows. Labels on arrows represent message names and may contain arguments. Activations (i.e., executing methods) are depicted by vertical rectangles. Actors are shown as the leftmost column.
Sequence diagrams can be used to describe either an abstract sequence (i.e., all possible interactions) or concrete sequences (i.e., one possible interaction, as in Figure 2.10). When describing all possible interactions, sequence diagrams also provide notations for conditionals and
iterators.

Software Engineering module | PDF page 51 | lines 14-24

2.3.2. Class Diagrams
Classes and objects
Class diagrams describe the structure of the system in terms of classes and objects. Classes are abstractions that specify the attributes and behavior of a set of objects. Objects are entities that encapsulate state and behavior. Each object has an identity: It can be referred individually and is distinguishable from other objects.
In UML, classes and objects are depicted by boxes including three compartments. The top compartment displays the name of the class or object. The center compartment displays its attributes, and the bottom compartment displays its operations. The attribute and operation compartment can be omitted for clarity. Object names are underlined to indicate that they are instances. By convention, class names start with an uppercase letter. Objects in object diagrams

Question sources

Computer Science model exam · 2015 · Question 41

Computer Science model exam · 2015 · Question 126

Question 56 | Q00575

Requirements and technical feasibility are unclear. The immediate goal is to build an exploratory version for user feedback and feasibility learning, then discard it rather than deliver it as production software. Which model best names that immediate activity?

A. Incremental model

B. Prototype model

C. Spiral model

D. Waterfall model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Prototype model

A throwaway prototype explores requirements, usability or technical feasibility and is discarded after learning. That exactly matches the immediate purpose specified. Incremental delivery would provide usable production capability; spiral would organize broader commitments around risk-driven cycles and might use a prototype within them. Classical waterfall favors understood requirements and familiar technology. The specific learning-and-discard goal therefore identifies the prototype model.

Why each choice is right or wrong

A - Incorrect: Incremental delivery targets usable capability rather than a discarded exploratory artifact.

B - Correct: A throwaway prototype explores needs and feasibility before being discarded.

C - Incorrect: Spiral may use a prototype, but names the broader risk-driven process.

D - Incorrect: The stated uncertainty lacks classical waterfall's stable, familiar conditions.

Primary note proof

Study-note section SE07

> A prototype explores requirements, usability or technical feasibility. A throwaway prototype is discarded after learning; an evolutionary prototype is deliberately refined toward production. A quick prototype may omit robustness, security or architecture, so do not deploy it unchanged without evaluation. Coding may occur during prototyping even if a diagram labels that activity construction rather than coding.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 26 | lines 6-9

> 2. Requirement's elicitation and analysis: This is the process of deriving the system requirements through observation of existing systems, discussions with potential users and procurers, task analysis, and so on. This may involve the development of one or more system models and prototypes. These help you understand the system to be specified.

Software Engineering module | PDF page 82 | lines 23-29

> • Prototype test. During this type of test, one or more users are presented with a piece of software that practically implements the system. A vertical prototype implements a complete use case through the system, and a horizontal prototype presents an interface for most use cases (without providing much or any functionality). The advantages of prototype tests are that they provide a realistic view of the system to the user and that prototypes can be instrumented to collect detailed data. The disadvantages of prototypes are that they are expensive to build and to modify.

Software Engineering module | PDF page 20 | lines 18-22

> the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
> In principle, the waterfall model should only be used when the requirements are well understood
> and unlikely to change radically during system development. However, the waterfall model

Question sources

Computer Science model exam · 2015 · Question 44

Question 57 | Q00580

What is the purpose of a sequence diagram?

A. To show the interactions between actors and a system

B. To show the behavior of objects in a system

C. To show the flow of activities in a system

D. To show the order and timing of interactions between objects

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer D: To show the order and timing of interactions between objects

The vertical axis is time and arrows are messages among participating objects; this directly supports order and timing.

Why each choice is right or wrong

A - Incorrect: Actor interaction is part of some sequences but is less precise than object-message order. See PDF p. 54.

B - Incorrect: Broadly describes behavior without identifying temporal interaction. See PDF p. 54.

C - Incorrect: Activity flow is represented by activity diagrams. See PDF p. 54.

D - Correct: Matches time-ordered messages and activations. See PDF p. 54.

Primary note proof

Study-note section SE19

> Sequence diagrams show interactions across lifelines, with order/time generally downward. Messages cross between participants; activation rectangles show execution. A creation message starts a lifeline; an X marks modeled destruction, not a guarantee about when a garbage collector runs. Synchronous calls wait for completion; asynchronous messages permit the sender to proceed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 54 | lines 26-35

> both buttons simultaneously. Figure 2.10 depicts a sequence diagram for an actor setting his 2Bwatch one minute ahead.
> Each column represents an object that participates in the interaction. The vertical axis represents
> time from top to bottom. Messages are shown by arrows. Labels on arrows represent message names and may contain arguments. Activations (i.e., executing methods) are depicted by vertical rectangles. Actors are shown as the leftmost column.
> Sequence diagrams can be used to describe either an abstract sequence (i.e., all possible interactions) or concrete sequences (i.e., one possible interaction, as in Figure 2.10). When describing all possible interactions, sequence diagrams also provide notations for conditionals and
> iterators.

Question sources

Computer Science model exam · 2015 · Question 49

Question 58 | Q00592

Which of the following diagram types is used to represent the static structure of a system?

A. Use case documentation

B. Sequence diagram

C. Class diagram

D. Activity diagram

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Class diagram

Class diagrams describe system structure in classes, attributes, operations and relationships; sequence and activity models describe behavior.

Why each choice is right or wrong

A - Incorrect: Documents functions from actors' viewpoints. See PDF p. 51, 101.

B - Incorrect: Represents interactions over time. See PDF p. 51, 101.

C - Correct: Explicit structural description of classes and objects. See PDF p. 51, 101.

D - Incorrect: Models activities and control flow. See PDF p. 51, 101.

Primary note proof

Study-note section SE17

> A class box lists name, attributes and operations; + public, - private, # protected and ~ package indicate visibility. A class diagram shows static types and relationships, while an object diagram shows particular instances/links. UML is maintained and standardized by OMG; it is a modeling language, not a lifecycle process. Structural things describe conceptual or physical structural elements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 51 | lines 14-24

> 2.3.2. Class Diagrams
> Classes and objects
> Class diagrams describe the structure of the system in terms of classes and objects. Classes are abstractions that specify the attributes and behavior of a set of objects. Objects are entities that encapsulate state and behavior. Each object has an identity: It can be referred individually and is distinguishable from other objects.
> In UML, classes and objects are depicted by boxes including three compartments. The top compartment displays the name of the class or object. The center compartment displays its attributes, and the bottom compartment displays its operations. The attribute and operation compartment can be omitted for clarity. Object names are underlined to indicate that they are instances. By convention, class names start with an uppercase letter. Objects in object diagrams

Software Engineering module | PDF page 101 | lines 5-6

> In UML, the functional model is represented with use case diagrams, the object model with class diagrams, and the dynamic model with statechart and sequence diagrams.

Question sources

Computer Science model exam · 2015 · Question 61

Question 59 | Q00597

Which is the following is the most important feature of the spiral model?

A. Efficiency management

B. Time management

C. Quality management

D. Risk management

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer D: Risk management.

SE06 makes risk the organizing principle of spiral development. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops an appropriate next result and plans further commitment. Time, efficiency and quality still matter, but they do not distinguish spiral from other processes as directly as its explicit risk-driven cycles.

Why each choice is right or wrong

A - Incorrect: Efficiency is useful, but it is not the defining organizing feature of spiral cycles.

B - Incorrect: Time management matters in many processes and does not distinguish spiral.

C - Incorrect: Quality management is important generally, but the defining spiral emphasis is risk.

D - Correct: Each spiral cycle explicitly evaluates and reduces the most important risks.

Primary note proof

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 96 | lines 12-21

> 4.7. Risk Management
> • Risk management is concerned with identifying risks and drawing up plans to minimise their effect on a project.
> • A risk is a probability that some adverse circumstance will occur
> - Project risks affect schedule or resources;
> - Product risks affect the quality or performance of the software being developed;
> - Business risks affect the organisation developing or procuring the software.

Risk Management Process
- Risk identification- Identify project, product and business risks;
- Risk analysis -Assess the likelihood and consequences of these risks;

Software Engineering module | PDF page 121 | lines 6-15

6.1.4. Software architecture
As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept
of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
In this section, we describe a few sample architectures that can be used for different systems. This
is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
representative examples and refer the reader to the literature for more details.

Question sources

Computer Science model exam · 2015 · Question 66

Question 60 | Q00606

_____ is the process of writing down the user and system requirements in a requirements document.

A. Feasibility study

B. Requirements elicitation and analysis.

C. Requirements specification

D. Requirements validation

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Requirements specification

Specification translates analyzed information into a document containing user requirements and detailed system requirements.

Why each choice is right or wrong

A - Incorrect: Decides viability before detailed analysis. See PDF p. 26, 25, 26.

B - Incorrect: Derives requirements rather than being the document-writing step. See PDF p. 26, 25, 26.

C - Correct: Directly names translation into the requirements document. See PDF p. 26, 25, 26.

D - Incorrect: Checks realism, consistency and completeness of the document. See PDF p. 26, 25, 26.

Primary note proof

Study-note section SE08

> Business requirements explain why the project exists and its goals. User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints. The Software Requirements Specification (SRS) organizes the agreed requirements. A natural-language specification communicates with clients while analysis models formalize it for technical discussion; real specifications can also contain diagrams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 26 | lines 10-17

> 3. Requirement's specification: Requirements specification is the activity of translating the information gathered during the analysis activity in to a document that defines a set of requirements. Two types of requirements may be included in this document. User requirements are
> abstract statements of the system requirements for the customer and end-user of the system; system
> requirements are a more detailed description of the functionality to be provided.
> 4. Requirement's validation: This activity checks the requirements for realism, consistency, and completeness. During this process, errors in the requirements document are inevitably discovered.
> It must then be modified to correct these problems.

Software Engineering module | PDF page 25 | lines 15-18

> There are four main activities in the requirements engineering process:
> 1. Feasibility study: Feasibility shows viability and practicality of a software, project or idea. An estimate is made of whether the identified user needs may be satisfied using current software and hardware technologies. The study considers whether the proposed system will be cost-

Software Engineering module | PDF page 26 | lines 6-9

> 2. Requirement's elicitation and analysis: This is the process of deriving the system requirements through observation of existing systems, discussions with potential users and procurers, task analysis, and so on. This may involve the development of one or more system models and prototypes. These help you understand the system to be specified.

Question sources

Computer Science model exam · 2015 · Question 75

Question 61 | Q00612

Incremental model of software development is

A. A reasonable mode when requirements are well define

B. Good model when a working core product is required quickly

C. Good model for the projects with large development teams

D. Good model when user requirements are not complete

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Good model when a working core product is required quickly

Early increments are usable stripped-down products; early working capability is the clearest fit among these options.

Why each choice is right or wrong

A - Incorrect: Well-defined requirements can be useful, but are not the distinctive advantage asked. See PDF p. 22, 22.

B - Correct: Directly supported by early working software. See PDF p. 22, 22.

C - Incorrect: The module specifically notes fewer staff can implement early increments. See PDF p. 22, 22.

D - Incorrect: Feedback can refine requirements, but the module also calls for sufficient overall definition to partition them. See PDF p. 22, 22.

Primary note proof

Study-note section SE04

> Choose incremental delivery when prioritized partial capability must reach users early. Smaller changes can be easier to debug and permit feedback, though architecture and integration still require planning. Iteration need not rebuild the whole system every time, and increments need not be perfectly independent modules. Broad descriptions of early working software can fit prototypes, iterative or Agile processes; identify the purpose of that early version.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 22 | lines 8-12

> This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
> also provide a platform for evaluation by the user.

Software Engineering module | PDF page 22 | lines 24-29

Advantages of Incremental Model:
• Generates working software quickly and early during the software life cycle. This model is more flexible - less costly to change scope and requirements. It is easier to test and debug during a smaller iteration.
• In this model customer can respond to each built. Lowers initial delivery cost.
• Easier to manage risk because risky pieces are identified and handled during its iteration.

Question sources

Computer Science model exam · 2015 · Question 81

Question 62 | Q00620

Which relationship are used when you have one use case that is similar to another, but does something slightly differently or something more.

A. Generalization

B. Aggregation

C. Composition

D. Association

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer A: Generalization

Use-case generalization specializes a general use case by adding detail or varying its realization, as in password versus smart-card authentication.

Why each choice is right or wrong

A - Correct: Direct specialization mechanism described by the module. See PDF p. 51.

B - Incorrect: Whole-part relationship, not use-case specialization. See PDF p. 51.

C - Incorrect: Whole-part lifecycle ownership, not use-case specialization. See PDF p. 51.

D - Incorrect: Communication/linkage alone does not express the stated specialized behavior. See PDF p. 51.

Primary note proof

Study-note section SE16

Generalization relates a specialized actor/use case to a more general one, with a hollow triangle toward the general element. An abstract use case specifies behavior for specialization and is not directly instantiated as a complete use case. It is not defined merely by reuse or lack of a direct actor line. Included behavior is incorporated where the inclusion is reached; do not infer every conditional branch in the base always executes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 51 | lines 4-13

> Generalization relationships
> Generalization/specialization relationships are a third mechanism for reducing the complexity of a model. A use case can specialize a more general one by adding more detail. For example, FieldOfficers are required to authenticate before they can use FRIEND. During early stages of requirements elicitation, authentication is modeled as a high-level Authenticate use case. Later, developers describe the Authenticate use case in more detail and allow for several different hardware platforms. This refinement activity results in two more use cases, AuthenticateWithPasswordwhich enables FieldOfficers to login without any specific hardware, and AuthenticateWithCard, which enables FieldOfficers to login using a smart card. The two new use cases are represented as specializations of the Authenticate use case.

Question sources

Computer Science model exam · 2015 · Question 89

Question 63 | Q00649

_____a software product is necessary to rectify the bugs observed while the system is in use.

A. Adaptive maintenance

B. Corrective maintenance

C. Perfective maintenance

D. Preventive maintenance

Hint

Classify the motive for the change before its implementation: repair, environment, useful capability or reduced future risk.

Answer B: Corrective maintenance

The purpose of the change determines the maintenance category. Corrective maintenance repairs faults observed during use. Adaptive maintenance handles environmental changes, perfective maintenance improves useful capability or performance, and preventive maintenance reduces future risk before a corresponding failure occurs. Because the scenario starts with observed bugs that need repair, corrective maintenance is the direct category rather than one of the other three motives.

Why each choice is right or wrong

A - Incorrect: Adaptive maintenance responds to a changed environment.

B - Correct: Corrective maintenance repairs observed defects.

C - Incorrect: Perfective maintenance improves useful capability or performance.

D - Incorrect: Preventive maintenance reduces future risk rather than primarily fixing the stated observed bugs.

Primary note proof

Study-note section SE40

> Corrective maintenance fixes faults; adaptive maintenance responds to environment changes; perfective maintenance improves useful capability/performance; preventive maintenance reduces future risk, such as restructuring fragile code. Motive determines category; refactoring can support preventive or perfective work. Maintenance often lasts years and includes understanding legacy code and regression effort, so it may dominate lifecycle cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 19 | lines 24-29

> 5. Operation and maintenance: Normally (although not necessarily), this is the longest life cycle phase. Maintenance of a typical software product requires much more than the effort necessary to
> develop the product itself. The system is installed and put into practical use. Maintenance involves
> correcting errors which were not discovered in earlier stages of the life cycle, improving the implementation of system units and enhancing the system ‘s services as new requirements are discovered.

Question sources

Computer Science model exam · 2015 · Question 118

Question 64 | Q00677

Criteria that should not be considered when evaluating or assessing requirement analysis?

A. Correctness

B. Need assessment

C. Consistency

D. Speed

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer D: Speed

Validation checks correctness, consistency and realism against stakeholder needs. Speed is a property of the future system's performance, not a general criterion for whether the analysis is valid.

Why each choice is right or wrong

A - Incorrect: Explicit requirements-validation property. See PDF p. 68, 79.

B - Incorrect: Assessing actual needs supports faithful requirements. See PDF p. 68, 79.

C - Incorrect: Explicit absence-of-contradiction property. See PDF p. 68, 79.

D - Correct: System response speed is a nonfunctional requirement, not an analysis quality criterion itself. See PDF p. 68, 79.

Primary note proof

Study-note section SE11

> Correct requirements reflect actual stakeholder needs. Complete requirements address the agreed scope, normal and exceptional behavior. Consistency excludes contradictions. Clarity/unambiguity gives one intended interpretation; feasibility/realism means achievable within constraints. Verifiability means objective evidence can check fulfillment. Replace “fast for most users” with a defined metric, workload and threshold.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 68 | lines 18-26

> system specification. Requirement validation involves checking if the specification is correct, complete, consistent, unambiguous, and realistic. A specification is correct if it represents the client’s view of the system (i.e., everything in the requirements model accurately represents an aspect of the system). It is complete if all possible scenarios through the system are described, including exceptional behavior (i.e., all aspects of the system are represented in the requirements
> model). The system specification is consistent if it does not contradict itself. The system specification is unambiguous if exactly one system is defined (i.e., it is not possible to interpret the specification two or more different ways). Finally, it is realistic if the system can be implemented within constraints. These properties are illustrated with UML instance diagrams in

Software Engineering module | PDF page 79 | lines 3-9

> • Performance characteristics. How responsive should the system be? How many concurrent users should it support? What is a typical or extreme load?
> • Error handling and extreme conditions. How should the system handle exceptions? Which exceptions should the system handle? What is the worse environment in which the system is expected to perform? Are there safety requirements on the system?
> • Quality issues. How reliable/available/robust should the system be? What is the client’s involvement in assessing the quality of the system or the development process?

Question sources

Computer Science model exam 2 · Question 18

Question 65 | Q00739

A measurement that defines the level of inter-dependability among modules of a programis__.

A. Cohesive

B. Coupling

C. Design

D. Testing

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: Coupling

Coupling measures dependency between subsystems; coherence/cohesion concerns dependency within one subsystem.

Why each choice is right or wrong

A - Incorrect: Internal relatedness, not inter-module dependency. See PDF p. 120, 120.

B - Correct: Exact strength-of-dependencies measure between subsystems. See PDF p. 120, 120.

C - Incorrect: An activity that chooses structures, not this measurement. See PDF p. 120, 120.

D - Incorrect: An error-detection activity, not this measurement. See PDF p. 120, 120.

Primary note proof

Study-note section SE22

> Cohesion measures how well elements inside a module serve a related purpose; coupling measures dependency between modules. Seek high cohesion and low coupling. Shared mutable data and knowledge of another module's internal representation increase coupling. Information hiding exposes stable operations while concealing changeable decisions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 120 | lines 6-12

> 6.1.3. Coupling and coherence
> Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
> is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Software Engineering module | PDF page 120 | lines 24-27

Coherence is the strength of dependencies within a subsystem. If a subsystem contains many objects that are related to each other and perform similar tasks, its coherence is high. If a subsystem
contains a number of unrelated objects, its coherence is low. A desirable property of a subsystem decomposition is that it leads to subsystems with high coherence.

Question sources

Computer Science model exam 2 · Question 80

Question 66 | Q00740

Which are supported modularization benefits? Select all that apply. Read D as a claim that modularization alone guarantees security.

A. Smaller components are difficult to maintain

B. Components with high cohesion can be re-used again

C. Program can be divided based on functional aspects

D. Modularization alone guarantees security.

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B, C: Components with high cohesion can be re-used again; Program can be divided based on functional aspects

Cohesive components with clear interfaces can be reused, and functional decomposition divides responsibilities into manageable modules. Smaller modules can also improve understanding and maintenance, so A reverses the ordinary benefit. Security requires appropriate protection and boundary enforcement; simply dividing code into modules does not guarantee it. Thus B and C are supported structural benefits without the unconditional security promise.

Why each choice is right or wrong

A - Incorrect: This reverses the ordinary maintenance benefit of manageable cohesive modules.

B - Correct: Cohesive components can be reused in suitable contexts.

C - Correct: Functional decomposition separates related responsibilities into modules.

D - Incorrect: Module boundaries alone do not guarantee security.

Primary note proof

Study-note section SE22

Logical cohesion groups similar-category operations, often selected by a flag; temporal cohesion groups operations done at the same time; functional cohesion focuses on one task. Splitting an unrelated three-purpose component improves modularity. If behavior is preserved, the restructuring is refactoring; the words are not mutually exclusive answers. Smaller modules can

aid understanding, independent work, reuse and maintenance, but too many interfaces also create complexity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 118 | lines 11-16

6.1.1. Subsystems and classes
In order to reduce the complexity of the application domain, we identified smaller parts called classes and organized them into packages. Similarly, to reduce the complexity of the solution domain, we decompose a system into simpler parts, called subsystems, which are made of a number of solution domain classes. In the case of complex subsystems, we recursively apply this principle and decompose a subsystem into simpler subsystems (see Figure 6.2).

Software Engineering module | PDF page 120 | lines 24-27

Coherence is the strength of dependencies within a subsystem. If a subsystem contains many objects that are related to each other and perform similar tasks, its coherence is high. If a subsystem
contains a number of unrelated objects, its coherence is low. A desirable property of a subsystem decomposition is that it leads to subsystems with high coherence.

Software Engineering module | PDF page 32 | lines 12-21

A class in an object-oriented system carefully delineates between its interfaces the routines and attributes within a class are held together tightly. In a properly designed system, the classes will be grouped into subsystems but remain independent; therefore, changing one class has no impact
on other classes, and so, the impact is minimized. However, the object-oriented approach is not a panacea; nothing is magical here that will promote perfect design or perfect code.
Promotion of reusability
Objects are reusable because they are modeled directly out of a real-world problem domain. Each
object stands by itself or within a small circle of peers (other objects). Within this framework, the class does not concern itself with the rest of the system or how it is going to be used within a particular system.

Question sources

Computer Science model exam 2 · Question 81

Question 67 | Q00741

Which of the following is not element of Requirement Engineering Process?

A. Software Requirement Specification

B. Feasibility Steady

C. Software Requirement Design

D. Requirement Gathering

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Software Requirement Design

The four listed requirements-engineering activities are feasibility, elicitation/analysis, specification and validation. Software requirement design is not one of them.

Why each choice is right or wrong

A - Incorrect: Specification is explicitly included. See PDF p. 25, 26, 26.

B - Incorrect: Feasibility study is explicitly included, despite the typo Steady. See PDF p. 25, 26, 26.

C - Correct: Designing the solution is separate from the listed requirements-engineering activities. See PDF p. 25, 26, 26.

D - Incorrect: Gathering is part of elicitation/analysis. See PDF p. 25, 26, 26.

Primary note proof

Study-note section SE08

> Business requirements explain why the project exists and its goals. User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints. The Software Requirements Specification (SRS) organizes the agreed requirements. A natural-language specification communicates with clients while analysis models formalize it for technical discussion; real specifications can also contain diagrams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 25 | lines 15-18

> There are four main activities in the requirements engineering process:
> 1. Feasibility study: Feasibility shows viability and practicality of a software, project or idea. An estimate is made of whether the identified user needs may be satisfied using current software and hardware technologies. The study considers whether the proposed system will be cost-

Software Engineering module | PDF page 26 | lines 10-17

> 3. Requirement’s specification: Requirements specification is the activity of translating the information gathered during the analysis activity in to a document that defines a set of requirements. Two types of requirements may be included in this document. User requirements are
> abstract statements of the system requirements for the customer and end-user of the system; system
> requirements are a more detailed description of the functionality to be provided.
> 4. Requirement’s validation: This activity checks the requirements for realism, consistency, and completeness. During this process, errors in the requirements document are inevitably discovered.
> It must then be modified to correct these problems.

Software Engineering module | PDF page 26 | lines 6-9

> 2. Requirement’s elicitation and analysis: This is the process of deriving the system requirements through observation of existing systems, discussions with potential users and procurers, task analysis, and so on. This may involve the development of one or more system models and prototypes. These help you understand the system to be specified.

Question sources

Computer Science model exam 2 · Question 82

Question 68 | Q00742

Which type of test determine if the system meets the entire requirements (functional and nonfunctional)

A. Unit Test

B. Integration test

C. System test

D. Acceptance test

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer C: System test

System testing examines the complete system's functional and nonfunctional requirements. Acceptance is a client decision against the agreement, which may also cover requirements but is a more specific acceptance purpose.

Why each choice is right or wrong

A - Incorrect: Isolates objects/components. See PDF p. 144, 147, 145.

B - Incorrect: Finds faults among combined components. See PDF p. 144, 147, 145.

C - Correct: Explicit complete-system functional/nonfunctional scope. See PDF p. 144, 147, 145.

D - Incorrect: Client agreement testing, rather than the general full-system test scope asked. See PDF p. 144, 147, 145.

Primary note proof

Study-note section SE35

> Acceptance testing supports the customer/user’s acceptance decision against agreed needs and criteria. Alpha typically happens in a controlled developer setting with selected users; beta uses external users in their environments. None of these levels is universally the longest. Unit is a level, while white-box is a technique; a unit can be tested using black-box cases too.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Software Engineering module | PDF page 147 | lines 14-18

• Unit testing, which finds faults by isolating an individual component using test stubs and drivers and by exercising the component using a test case
• Integration testing, which finds faults by integrating several components together
• System testing, which focuses on the complete system, its functional and nonfunctional requirements, and its target environment

Software Engineering module | PDF page 145 | lines 3-5

o Acceptance testing and installation testing check the requirements against the project agreement and should be done by the client, if necessary with support by the developers.

Question sources

Computer Science model exam 2 · Question 83

Question 69 | Q00744

All of the following are required to develop an effective test except:

A. Detailed understanding of the system

B. Application and solution domain knowledge

C. Knowledge of the testing techniques

D. Skill to specific programming language

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer D: Skill to specific programming language

Black-box equivalence tests can be derived from inputs and expected behavior without writing in a particular implementation language. Domain/system knowledge and testing methods remain relevant.

Why each choice is right or wrong

A - Incorrect: Needed to choose meaningful cases and expected results. See PDF p. 149, 149.

B - Incorrect: Needed to interpret required behavior. See PDF p. 149, 149.

C - Incorrect: Needed to partition inputs or choose boundaries/paths appropriately. See PDF p. 149, 149.

D - Correct: Not universally necessary for behavior-based test design; source-language knowledge applies particularly to white-box work. See PDF p. 149, 149.

Primary note proof

Study-note section SE36

> Functional tests check required behavior against the SRS and acceptance criteria. Performance tests assess speed, throughput and resources under a specified workload; load uses expected demands and stress exceeds normal limits. Usability, security, compatibility and reliability tests target their respective qualities. “Normal operating conditions” alone does not distinguish functional from performance testing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 149 | lines 6-10

> Equivalence testing is a black box testing technique that minimizes the number of test cases. The possible inputs are partitioned into equivalence classes, and a test case is selected for each class. The assumption of equivalence testing is that systems usually behave in similar ways for all members of a class. To test the behavior associated with an equivalence class, we only need to test
> one member of the class. Equivalence testing consists of two steps: identification of the

Software Engineering module | PDF page 149 | lines 24-29

> Path testing
> Path testing is a white box testing technique that identifies faults in the implementation of the component. The assumption behind path testing is that, by exercising all possible paths through the code at least once, most faults will trigger failures. The identification of paths requires knowledge of the source code and data structures.
> The starting point for path testing is the flow graph. A flow graph consists of nodes representing

Question sources

Computer Science model exam 2 · Question 85

Question 70 | Q00745

Use case description commonly contain:

A. Trigger, associations, guarantee.

B. Precondition, generalizations, main scenario.

C. Precondition, main scenario, alternative scenarios

D. None of the above

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Precondition, main scenario, alternative scenarios

A use-case description includes entry conditions, the numbered normal flow and exceptional cases.

Why each choice is right or wrong

A - Incorrect: Trigger/associations/guarantee are not its stated three-part summary. See PDF p. 48.

B - Incorrect: Generalization is a relationship, not one of these template fields. See PDF p. 48.

C - Correct: Matches preconditions and normal/exceptional flow content. See PDF p. 48.

D - Incorrect: A supported summary is offered. See PDF p. 48.

Primary note proof

Study-note section SE15

> Example: Submit Incident. Precondition: officer authenticated. 1 Officer supplies location and description. 2 System validates details. 3 System records the incident. 4 System returns its ID. Alternative 2a: location missing → report the problem and request correction; save nothing. Success postcondition: incident and ID persist; failure guarantee: no partial record. An alternate flow is a conditional variation; an exception flow handles a failure. Preconditions hold before entry, while a trigger starts the use case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 48 | lines 18-32

> To describe a use case, we use a template composed of six fields (see also Figure 2.2):
> • The name of the use case is unique across the system so that developers (and project participants) can unambiguously refer to the use case.
> • Participating actors are actors interacting with the use case.

• Entry conditions describe the conditions that need to be satisfied before the use case is initiated.
• The flow of events describes the sequence of actions of the use case, which are be numbered for reference. The common case (i.e., cases that occur frequently) and the exceptional cases (i.e., cases that seldom occur, such as errors and unusual conditions) are described separately in different use cases for clarity.
• Exit conditions describe the conditions that are satisfied after the completion of the use case.
• Special requirements are requirements that are not related to the functionality of the system. These include constraints on the performance of the system, its implementation, the hardware platforms it runs on, and so on.

Question sources

Computer Science model exam 2 · Question 86

Question 71 | Q00746

Aggregation, composition, generalization. Which document is created by a system analyst after the requirements are collected from different stakeholders?

A. Requirement gathering

B. Software Requirement specification

C. Software Requirement testing

D. Software Requirement validation

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: Software Requirement specification

After collection, requirements are organized into the SRS. The opening aggregation/composition/generalization fragment does not alter the requested document.

Why each choice is right or wrong

A - Incorrect: A gathering activity, not the resulting specification document. See PDF p. 26, 83.

B - Correct: The defined organized requirements document. See PDF p. 26, 83.

C - Incorrect: Testing is an activity rather than this document. See PDF p. 26, 83.

D - Incorrect: Validation checks a specification rather than naming its document. See PDF p. 26, 83.

Primary note proof

Study-note section SE08

Business requirements explain why the project exists and its goals. User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints. The Software Requirements Specification (SRS) organizes the agreed requirements. A natural-language specification communicates with clients while analysis models formalize it for technical discussion; real specifications can also contain diagrams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 26 | lines 10-17

3. Requirement's specification: Requirements specification is the activity of translating the information gathered during the analysis activity in to a document that defines a set of requirements. Two types of requirements may be included in this document. User requirements are
abstract statements of the system requirements for the customer and end-user of the system; system
requirements are a more detailed description of the functionality to be provided.
4. Requirement's validation: This activity checks the requirements for realism, consistency, and completeness. During this process, errors in the requirements document are inevitably discovered.
It must then be modified to correct these problems.

Software Engineering module | PDF page 83 | lines 12-25

3.4.4. Documenting Requirements Elicitation
The results of the requirements elicitation activity and the analysis activity are documented in the
Requirements Analysis Document (RAD). This document completely describes the system in terms of functional and nonfunctional requirements and serves as a contractual basis between the
client and the developers. The audience for the RAD includes the client, the users, the project management, the system analysts (i.e., the developers who participate in the requirements), and the system designers (i.e., the developers who participate in the system design). The first part of the document, including use cases and nonfunctional requirements, is written during requirements
elicitation. The formalization of the specification in terms of object models is written during analysis.
The first section of the RAD is an Introduction. Its purpose is to provide a brief overview of the function of the system and the reasons for its development, its scope, and references to the development context (e.g., related problem statement, references to existing systems, feasibility studies). The introduction also includes the objectives and success criteria of the project.

Question sources

Computer Science model exam 2 · Question 87

Question 72 | Q00749

_____is a process which translate system specification into a formal or semi formal model.

A. Implementation

B. Elicitation

C. Analysis

D. Design

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Analysis

Analysis structures and formalizes the natural-language system specification into a formal or semiformal analysis model.

Why each choice is right or wrong

A - Incorrect: Converts design to executable code. See PDF p. 100, 62.

B - Incorrect: Produces the initial specification with stakeholders. See PDF p. 100, 62.

C - Correct: Exact formalization activity described. See PDF p. 100, 62.

D - Incorrect: Derives internal solution structure after analysis. See PDF p. 100, 62.

Primary note proof

Study-note section SE13

> A scenario is one concrete execution; a use case abstracts the related successful and alternative scenarios. Analysis integrates functional (use cases/scenarios), structural/object (classes) and dynamic (states/interactions) models. Entity objects represent domain information such as Incident; boundary objects mediate interaction, such as ReportForm; control objects coordinate a use case. A UI form is normally boundary, not entity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 100 | lines 11-24

> Analysis focuses on producing a model of the system, called the analysis model, which is correct, complete, consistent, and verifiable. Analysis is different from requirements elicitation in that developers focus on structuring and formalizing the requirements elicited from users
> Figure 4.1 Products of requirements elicitation and analysis (UML activity diagram).
> Althoughtheanalysismodelmaynotbeunderstandabletotheusersandtheclient, it helps developers verify the system specification produced during requirements elicitation. There is a natural tendency for users and developers to postpone difficult decisions until later in the project. A decision may be difficult because of lack of domain knowledge, lack of technological knowledge,

or simply because of disagreements among users and developers. Postponing decisions enables the
project to move on smoothly and avoid confrontation with reality or peers. Unfortunately, difficult
decisions will eventually need to be made, often at higher cost when intrinsic problems are discovered during testing, or worse, during user evaluation. Translating a system specification in to a formal or semi-formal model forces developers to identify and resolve difficult issues early in
the development.

Software Engineering module | PDF page 62 | lines 9-16

Requirement's elicitation focuses on describing the purpose of the system. The client, the developers, and the users identify a problem area and define a system that addresses the problem.
Such a definition is called a system specification and serves as a contract between the client and the developers. The system specification is structured and formalized during analysis to produce an analysis model (see Figure 3.1). Both system specification and analysis model represent the same information. They differ only in the language and notation they use. The system specification
is written in natural language, whereas the analysis model is usually expressed in a formal or semiformal notation.

Question sources

Computer Science model exam 2 · Question 90

Question 73 | Q00750

Which maintenance risk can arise from poorly designed objects with complex or poorly separated responsibilities?

A. Easier maintenance

B. Objects may be understood as stand-alone entities

C. Objects are potentially reusable components

D. Poorly designed objects can be difficult to maintain.

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: Poorly designed objects can be difficult to maintain.

Encapsulation and reuse can ease maintenance when objects have clear responsibilities and interfaces. Poor object design can still be complex, so its objects may become difficult to maintain. That conditional risk is D. The other choices describe potential advantages of well-designed objects rather than disadvantages arising from the stated poor design. Object orientation alone does not guarantee either maintenance outcome.

Why each choice is right or wrong

A - Incorrect: Easier maintenance is a potential benefit, not this risk.

B - Incorrect: Understanding encapsulated objects through their interfaces is a potential benefit.

C - Incorrect: Potential reuse is an advantage of suitable design.

D - Correct: Poorly designed complex objects can be difficult to maintain.

Primary note proof

Study-note section SE22

> Cohesion measures how well elements inside a module serve a related purpose; coupling measures dependency between modules. Seek high cohesion and low coupling. Shared mutable data and knowledge of another module's internal representation increase coupling. Information hiding exposes stable operations while concealing changeable decisions.

Study-note section SE01

> Procedural decomposition organizes functions; object oriented decomposition organizes objects with state and behavior. Either can use iterative or sequential development. OOP does not automatically shorten every project. Reuse, encapsulation and consistent models can reduce duplication and ease change; poor object design can still be complex. Functional programming emphasizes function evaluation and controlled side effects; event-driven programs react to events.

Study-note section SE22

> Splitting an unrelated three-purpose component improves modularity. If behavior is preserved, the restructuring is refactoring; the words are not mutually exclusive answers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 31 | lines 19-31

> 1.3.1. Why an object oriented
> • Object oriented systems are
> o Easier to adapt to changes
> o More robust
> o Easier to maintain
> o Promote greater design and code reuse
> o Creates modules of functionality
> • REASONS FOR WORKING OF OBJECT ORIENTED SYSTEMS:

Higher level of abstraction
The object-oriented approach supports abstraction at the object level. Since objects encapsulate both data (attributes) and functions (methods), they work at a higher level of abstraction. The development can proceed at the object level and ignore the rest of the system for as long as necessary. This makes designing, coding, testing, and maintaining the system much simpler.

Question sources

Computer Science model exam 2 · Question 91

Question 74 | Q00897

In a class diagram, a filled diamond is at Car's end of its Engine association. In this model an Engine remains owned by its Car for its entire lifetime, with no detachment or transfer. What does the relationship imply?

A. Car uses Engine temporarily

B. Engine can exist independently of the Car

C. Engine is part of Car and cannot exist independently under the stated lifetime policy.

D. Car inherits from Engine

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Engine is part of Car and cannot exist independently under the stated lifetime policy.

The filled diamond at Car denotes composition: Car is the whole and Engine is its strongly owned part. Composition destroys parts that remain owned when the whole is destroyed. Because this model forbids detachment or transfer, the Engine's lifetime cannot continue independently of its Car. Temporary use would be a dependency, and inheritance would use a hollow generalization triangle instead.

Why each choice is right or wrong

A - Incorrect: Temporary use is a dependency, not composition.

B - Incorrect: The stipulated permanent ownership excludes an independent Engine lifetime.

C - Correct: Engine is the owned part and cannot outlive Car under the stated policy.

D - Incorrect: Inheritance uses a hollow triangle rather than a filled diamond.

Primary note proof

Study-note section SE18

Composition uses a filled diamond at the whole end and strong ownership: a part is in at most one composite at a time, and parts still owned are destroyed with that composite. Parts may be

detached/transferred under a defined model. Thus “cannot ever exist independently” is an oversimplification. For Car composing Engine, the diamond belongs beside Car; saying it points to Engine is imprecise. The Car-Engine example describes composition.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 51 | lines 14-24

2.3.2. Class Diagrams
Classes and objects
Class diagrams describe the structure of the system in terms of classes and objects. Classes are abstractions that specify the attributes and behavior of a set of objects. Objects are entities that encapsulate state and behavior. Each object has an identity: It can be referred individually and is distinguishable from other objects.
In UML, classes and objects are depicted by boxes including three compartments. The top compartment displays the name of the class or object. The center compartment displays its attributes, and the bottom compartment displays its operations. The attribute and operation compartment can be omitted for clarity. Object names are underlined to indicate that they are instances. By convention, class names start with an uppercase letter. Objects in object diagrams

Software Engineering module | PDF page 53 | lines 19-28

Aggregation
Associations are used to represent a wide range of connections among a set of objects. In practice,
a special case of association occurs frequently: composition. For example, a State contains many Counties, which in turn contains many Townships. A PoliceStation is composed of PoliceOfficers. Another example is a Directory that contains a number of Files. Such relationships could be modeled using a one-to-many association. Instead, UML provides the concept of an aggregation to denote composition. An aggregation is denoted by a simple line with a diamond at the container
end of the association. Although one-to-many associations and aggregations can be used alternatively, aggregations are preferable because they emphasize the hierarchical aspects of the relationship.

Question sources

Computer Science Exit Exam · 2018 · Question 10

Question 75 | Q00942

From the following statement which one is disadvantage of waterfall model?

A. model is simple and easy to understand and use

B. It is easy manage due to the rigidity of the model

C. .
Not suitable for projects where requirements are at a moderate to high risk of changing

D. In this model phases are processed and completed one at a time

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: .
Not suitable for projects where requirements are at a moderate to high risk of changing

Waterfall requires early stable requirements and has difficulty responding to change, so moderate-to-high change risk is its documented disadvantage.

Why each choice is right or wrong

A - Incorrect: Listed as an advantage. See PDF p. 20.

B - Incorrect: Listed as an advantage from distinct deliverables/reviews. See PDF p. 20.

C - Correct: Directly matches the disadvantage on PDF p.21. See PDF p. 20.

D - Incorrect: Sequential non-overlap is a structural feature, not the stated change-risk disadvantage. See PDF p. 20.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 20 | lines 18-22

> the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
> In principle, the waterfall model should only be used when the requirements are well

understood
and unlikely to change radically during system development. However, the waterfall model

Question sources

Computer Science Exit Exam · 2018 · Question 55

Question 76 | Q00948

Which statement specifically constrains response-time performance under a stated workload?

A. The system allow user to generate PDF report.

B. The system respond to user's request within 2 second under normal load

C. .
The system allow user to store data on relational database

D. The system allow user to submit emergency report

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: The system respond to user's request within 2 second under normal load

The two-second response obligation bounds the speed of a service under the stated load, so it is a performance requirement. Generating a PDF and submitting an emergency report describe services the system provides. The database clause describes storage behavior together with an implementation choice; it does not impose a response-time threshold. The explicit performance scope removes the broader taxonomy overlap between nonfunctional requirements and technology constraints.

Why each choice is right or wrong

A - Incorrect: Report generation specifies a service, not its response-time limit.

B - Correct: A two-second response bound under normal load is a quantitative speed target.

C - Incorrect: This specifies storage behavior and a database technology, without a timing bound.

D - Incorrect: Submitting an emergency report specifies a function rather than its speed.

Primary note proof

Study-note section SE10

Nonfunctional requirements include quality attributes and constraints. Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning. Security concerns protection; compatibility concerns cooperating environments; portability concerns moving environments; maintainability/supportability concern change, diagnosis and evolution.

Study-note section SE10

"Search responds within two seconds under 100 concurrent users" is measurable performance. "New users finish checkout within one minute without help" is usability. "Use Java" is a technology/pseudo-requirement, often separately classified as a system constraint. Dependability includes reliability, safety and robustness; portability is not its usual central category. Performance degradation at peak load primarily concerns performance, not automatically reliability.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 65 | lines 19-23

3.2.1. Functional Requirements
Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. For example, the following is an example of functional requirements for SatWatch, a watch that resets itself without user intervention:

Software Engineering module | PDF page 66 | lines 14-18

3.2.2. Non-Functional and Pseudo-Requirements
Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Nonfunctional requirements include quantitative constraints, such as response time (i.e., how fast the system reacts to user commands)
or accuracy (i.e., how precise are the system's numerical answers). The following are the

Software Engineering module | PDF page 67 | lines 3-8

Pseudo-requirements are requirements imposed by the client that restrict the implementation of the system. Typical pseudo-requirements are the implementation language and the platform on which the system is to be implemented. For life-critical developments, pseudo-requirements often
include process and documentation requirements (e.g., the use of a formal specification method, the complete release of all work products). Pseudo-requirements have usually no direct effect on the users' view of the system. The following are the pseudo functional requirements for SatWatch:

Question sources

Computer Science Exit Exam · 2018 · Question 61

Question 77 | Q00949

In object design, what does the contract typically includes?

A. The class diagram and sequence diagram

B. Project budget and timeline

C. Invariants, preconditions, post conditions

D. The use case names and actors role

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Invariants, preconditions, post conditions

Contracts establish class-wide invariants and caller/callee obligations before and after an operation.

Why each choice is right or wrong

A - Incorrect: Diagrams can document a design but are not the contract constraint categories. See PDF p. 136.

B - Incorrect: Project planning terms, not class contracts. See PDF p. 136.

C - Correct: Exactly the three contract constraint types. See PDF p. 136.

D - Incorrect: Use-case identifiers and roles do not specify class invariants and operation obligations. See PDF p. 136.

Primary note proof

Study-note section SE25

> A contract states preconditions, postconditions and invariants. For withdraw(amount): require amount>0 and amount<=balance; ensure balance=oldBalance-amount; preserve balance>=0. Validation checks inputs; implementation enforces the transition. A complete feature needs both UI/API and persisted behavior, not only a diagram.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 136 | lines 3-11

> • An invariant is a predicate that is always true for all instances of a class. Invariants are constraints associated with classes or interfaces. Invariants are used to specify consistency constraints among class attributes.
> • A precondition is a predicate that must be true before an operation is invoked. Preconditions are associated with a specific operation. Preconditions are used to specify constraints that a caller must meet before calling an operation.
> • A post-condition is a predicate that must be true after an operation is invoked. Post-conditions are associated with a specific operation. Post-conditions are used to specify constraints that the object must ensure after the invocation of the operation.

Question sources

Computer Science Exit Exam · 2018 · Question 62

Question 78 | Q00967

Which one of the following UML building block that defines the static part of the model and represents physical and conceptual elements?

A. Structural things

B. Behavioral things

C. ¢.
An notational things

D. Grouping things

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer A: Structural things

Structural things explicitly define the static model and physical/conceptual elements.

Why each choice is right or wrong

A - Correct: Matches the module verbatim. See PDF p. 45, 45, 46.

B - Incorrect: Behavioral things are dynamic. See PDF p. 45, 45, 46.

C - Incorrect: Annotational things capture notes and remarks. See PDF p. 45, 45, 46.

D - Incorrect: Grouping things organize elements into packages. See PDF p. 45, 45, 46.

Primary note proof

Study-note section SE17

> A class box lists name, attributes and operations; + public, - private, # protected and ~ package indicate visibility. A class diagram shows static types and relationships, while an object diagram shows particular instances/links. UML is maintained and standardized by OMG; it is a modeling language, not a lifecycle process. Structural things describe conceptual or physical structural elements.

Study-note section SE-S03

> In the course's UML building-block terminology, structural things describe static conceptual or physical elements. Behavioral things describe dynamic interactions and state behavior. Grouping things organize model elements into packages. Annotational things attach explanatory remarks or constraints, usually as notes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 45 | lines 3-17

> Structural things:
> The Structural things define the static part of the model. They represent physical and conceptual

elements. Following are the brief descriptions of the structural things.
Class:
Class represents set of objects having similar responsibilities.
Interface:
Interface defines a set of operations which specify the responsibility of a class.
Collaboration:
Collaboration defines interaction between elements.
Use case:
Use case represents a set of actions performed by a system for a specific goal.
Component:
Component describes physical part of a system.
Node:
A node can be defined as a physical element that exists at run time.

Software Engineering module | PDF page 45 | lines 18-23

Behavioral things:
A behavioral thing consists of the dynamic parts of UML models. Following are the behavioral things:
Interaction:
Interaction is defined as a behavior that consists of a group of messages exchanged among elements to accomplish a specific task.

Software Engineering module | PDF page 46 | lines 7-16

Grouping things:
Grouping things can be defined as a mechanism to group elements of a UML model together. There is only one grouping thing available:
Package:
Package is the only one grouping thing available for gathering structural and behavioral things.
An notational things:
An notational things can be defined as a mechanism to capture remarks, descriptions, and comments of UML model elements. Note is the only one An notational thing available.
Note:
A note is used to render comments, constraints etc. of an UML element.

Question sources

Computer Science Exit Exam · 2018 · Question 80

Question 79 | Q00971

After deploying a new software system, users reports that the system slows during peak hour. Which software quality attribute primarily affected?

A. Maintainability

B. Performance

C. Reliability

D. Usability

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: Performance

Slow response at peak load concerns responsiveness and concurrency/load constraints, hence performance.

Why each choice is right or wrong

A - Incorrect: Concerns cost/ease of modifying the system. See PDF p. 79, 66.

B - Correct: Directly affected response-speed/load quality. See PDF p. 79, 66.

C - Incorrect: Reliability concerns conformance/failures; slow service could violate requirements, but performance is the specific attribute. See PDF p. 79, 66.

D - Incorrect: Usability concerns ease of learning/using rather than primarily load-dependent response speed. See PDF p. 79, 66.

Primary note proof

Study-note section SE10

> “Search responds within two seconds under 100 concurrent users” is measurable performance. “New users finish checkout within one minute without help” is usability. “Use Java” is a technology/pseudo-requirement, often separately classified as a system constraint. Dependability includes reliability, safety and robustness; portability is not its usual central category. Performance degradation at peak load primarily concerns performance, not automatically reliability.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 79 | lines 3-9

> • Performance characteristics. How responsive should the system be? How many concurrent users should it support? What is a typical or extreme load?
> • Error handling and extreme conditions. How should the system handle exceptions? Which exceptions should the system handle? What is the worse environment in which the system is expected to perform? Are there safety requirements on the system?

• Quality issues. How reliable/available/robust should the system be? What is the client's involvement in assessing the quality of the system or the development process?

Software Engineering module | PDF page 66 | lines 14-18

3.2.2. Non-Functional and Pseudo-Requirements
Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Nonfunctional requirements include quantitative constraints, such as response time (i.e., how fast the system reacts to user commands)
or accuracy (i.e., how precise are the system's numerical answers). The following are the

Question sources

Computer Science Exit Exam · 2018 · Question 84

Question 80 | Q01197

Which choice most completely lists candidate heuristics for grouping objects into subsystems? Treat them as starting guidance, not mandatory boundaries.

A. Assign objects identified in one use case into the same subsystem.

B. Create a dedicated subsystem for objects used for moving data among subsystems.

C. Minimize the number of associations crossing subsystem boundaries.

D. All of the above

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: All of the above

SE21 recommends grouping related functionality, minimizing cross-subsystem associations and isolating communication infrastructure where useful. Grouping objects serving one use case can be a starting heuristic, though workflows do not dictate every boundary. B and C directly support isolation and low coupling. Since the question asks what can be a heuristic, the inclusive option is appropriate.

Why each choice is right or wrong

A - Incorrect: True as one component, but incomplete as the requested full selection. Objects collaborating on one use case can be grouped as a starting heuristic, subject to other design constraints.

B - Incorrect: True as one component, but incomplete as the requested full selection. Communication infrastructure can be isolated in a dedicated subsystem when useful.

C - Incorrect: True as one component, but incomplete as the requested full selection. Reducing cross-boundary associations reduces coupling between subsystems.

D - Correct: All three can guide decomposition, provided use-case grouping is not treated as an absolute rule.

Primary note proof

Study-note section SE21

> System design turns the analysis model into architecture: design goals, subsystem responsibilities, dependencies, hardware mapping, data management, access control, control flow and startup/shutdown/error policies. Detail/object design refines APIs, parameter/return types, exceptions, component choices and algorithms. A subsystem is a manageable logical collection with services and an interface; it need not be a physical replaceable artifact.

Study-note section SE21

> Group closely related objects and functionality, minimize associations crossing boundaries, and isolate communication infrastructure where useful.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 118 | lines 11-16

> 6.1.1. Subsystems and classes
> In order to reduce the complexity of the application domain, we identified smaller parts called classes and organized them into packages. Similarly, to reduce the complexity of the solution domain, we decompose a system into simpler parts, called subsystems, which are made of a number of solution domain classes. In the case of complex subsystems, we recursively apply this principle and decompose a subsystem into simpler subsystems (see Figure 6.2).

Software Engineering module | PDF page 116 | lines 16-29

> The analysis model describes the system completely from the actors' point of view and serves as the basis of communication between the client and the developers. The analysis model, however, does not contain information about the internal structure of the system, its hardware configuration,
> or, more generally, how the system should be realized. System design is the first step in this direction. System design results in the following products:
> • List of design goals, describing the qualities of the system that developers should optimize
> • Software architecture, describing the subsystem decomposition in terms of subsystem responsibilities, dependencies among subsystems, subsystem mapping to hardware, and major policy decisions such as control flow, access control, and data storage.
> The design goals are derived from the nonfunctional requirements. Design goals guide the decisions to be made by the developers especially when trade-offs are needed. The subsystem decomposition constitutes the bulk of system design. Developers divide the system into manageable pieces to deal with complexity: Each subsystem is assigned to a team and realized independently. In order for this to be possible, though, developers need to address system-wide

Software Engineering module | PDF page 120 | lines 6-12

6.1.3. Coupling and coherence
Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Question sources

Computer Science model exam 2 · Question 18

Question 81 | Q01205

Which statement is false about a logical software subsystem? Interpret complexity reduction as managing solution-design complexity, not changing the inherent application domain.

A. A physical and replaceable part of the system.

B. It helps manage the complexity of the solution design.

C. Decompose a system into simpler parts.

D. Well-defined interfaces that encapsulate the state and behavior of its contained classes.

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer A: A physical and replaceable part of the system.

A subsystem is a manageable logical collection of responsibilities with services and an interface. It need not be a physical replaceable artifact, so A is false as a definition. Decomposing the designed solution into simpler groups helps manage complexity, and defined interfaces can encapsulate implementation state and behavior. These design benefits concern the solution organization rather than reducing the inherent complexity of its problem domain.

Why each choice is right or wrong

A - Correct: A logical subsystem need not be physical or independently replaceable.

B - Incorrect: Managing solution-design complexity is a supported decomposition benefit.

C - Incorrect: Decomposition groups the system into simpler responsibilities.

D - Incorrect: Defined interfaces encapsulate contained implementation details.

Primary note proof

Study-note section SE21

Group closely related objects and functionality, minimize associations crossing boundaries, and isolate communication infrastructure where useful. User workflows inform decomposition but do not alone dictate the internal structure. Analysis describes domain needs; subsystem decomposition is principally a design result. Architecture documentation supports communication and records major decisions, not every source-level execution step.

Study-note section SE21

A subsystem is a manageable logical collection with services and an interface; it need not be a physical replaceable artifact.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 118 | lines 11-16

6.1.1. Subsystems and classes
In order to reduce the complexity of the application domain, we identified smaller parts called classes and organized them into packages. Similarly, to reduce the complexity of the solution domain, we decompose a system into simpler parts, called subsystems, which are made of a number of solution domain classes. In the case of complex subsystems, we recursively apply this principle and decompose a subsystem into simpler subsystems (see Figure 6.2).

Software Engineering module | PDF page 119 | lines 13-25

6.1.2. Services and subsystem interfaces
A subsystem is characterized by the services it provides to other subsystems. A service is a set of related operations that share a common purpose. A subsystem providing a notification service, for
example, defines operations to send notices, look up notification channels, and subscribe and unsubscribe to a channel.
The set of operations of a subsystem that are available to other subsystems form the subsystem interface. The subsystem interface, also referred to as the application programmer interface (API),
includes the name of the operations, their parameters, their types, and their return values. System
design focuses on defining the services provided by each subsystem that is, enumerating the operations, their parameters, and their high-level behavior. Object design will focus on defining the subsystem interfaces, that is, the type of the parameters and the return value of each operation.
The definition of a subsystem in terms of the services it provides helps us focus on its interface as opposed to its implementation. A good subsystem interface should provide as little information

Question sources

Computer Science model exam 2 · Question 26

Question 82 | Q01208

Which of the following
is an example of Usability requirements?

A. A new user shall complete the main interface task within one minute without help.

B. Robustness

C. Mean time to failure

D. Adaptability

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: A new user shall complete the main interface task within one minute without help.

A usability requirement states an observable target for successful and learnable use. Requiring a new user to finish a main interface task without help within one minute gives a task, user context, success condition and threshold. Robustness concerns dependable behavior under adverse conditions, mean time to failure is a reliability measure, and adaptability concerns change. Merely naming a user interface would not itself specify an obligation, so the option supplies a genuine requirement.

Why each choice is right or wrong

A - Correct: The task-success and unaided-learning target is a measurable usability obligation.

B - Incorrect: Robustness concerns dependable behavior under adverse conditions rather than ease of learning a task.

C - Incorrect: Mean time to failure measures reliability rather than user task ease.

D - Incorrect: Adaptability concerns accommodating change rather than the stated user task target.

Primary note proof

Study-note section SE10

> Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning.

Study-note section SE10

> “New users finish checkout within one minute without help” is usability.

Study-note section SE-S07

> Robustness concerns reliable operation across varied or difficult operating conditions and controlled failure outside the supported range. It is different from usability, which concerns

people successfully learning and carrying out tasks. Improving an interface's task success does not by itself establish robustness under adverse conditions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 78 | lines 12-30

3.3.7. Identifying Non-Functional Requirements
Nonfunctional requirements describe user-visible aspects of the system that are not directly related
to the functional behavior of the system. Nonfunctional requirements span a number of issues, from user interface look and feel to response time requirements to security issues.
Nonfunctional
requirements are defined at the same time as functional requirements are, because they have as much impact on the development and cost of the system.
For example, consider a mosaic display that an air traffic controller uses to track planes. A mosaic
display system complies data from a series of radar and databases (hence the term "mosaic") into
a summary display indicating all aircraft in a certain area, including their identification, speed, and
altitude. The number of aircraft such a system can display constrains the performance of the air traffic controller and the cost of the system. If the system can only handle a few aircraft simultaneously, the system cannot be used at busy airports. On the other hand, a system able to handle a large number of aircraft is more costly and more complex to build.
Nonfunctional requirements can be elicited by investigating the following issues.
• User interface and human factors. What kind of interface should the system provide? What is the level of expertise of the users?
• Documentation. What level of document is required? Should only user documentation be provided? Should there be technical documentation for maintainers? Should the development process be documented?

Software Engineering module | PDF page 82 | lines 3-12

though this type of approach could be used for any parameter, usability tests focus on usability parameters, such as the ease to learn the system, the time to accomplish a task, or the rate of errors
a user makes when accomplishing a task.
There are two important differences between classical experiments and usability tests. Whereas the classical experimental method is designed to confirm or refute a hypothesis, the goal of usability tests is to obtain qualitative information on how to fix usability problems and how to improve the system. Another difference is the rigor with which the experiments need to be performed. It has been shown that even a series of quick focused tests starting as early as requirements elicitation is extremely helpful. Nielsen uses the term "discount usability engineering" to indicate that a few usability tests are better than none at all.

Question sources

Computer Science model exam 2 · Question 29

Question 83 | Q01260

Which one of the following is a Nonfunctional requirement?

A. Admin can upload photos

B. Admin can search book

C. Librarian can delete student

D. The system delivers the search items within 2 seconds

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer D: The system delivers the search items within 2 seconds

The two-second limit constrains how quickly the search service responds; the others specify permitted operations.

Why each choice is right or wrong

A - Incorrect: Uploading is a service. See PDF p. 65, 66.

B - Incorrect: Searching is a service. See PDF p. 65, 66.

C - Incorrect: Deletion is a service. See PDF p. 65, 66.

D - Correct: Quantitative response-time requirement. See PDF p. 65, 66.

Primary note proof

Study-note section SE10

> Nonfunctional requirements include quality attributes and constraints. Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning. Security concerns protection; compatibility concerns cooperating environments; portability concerns moving environments; maintainability/supportability concern change, diagnosis and evolution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 65 | lines 19-23

> 3.2.1. Functional Requirements
> Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. For example, the following is an example of functional requirements for SatWatch, a watch that resets itself without user intervention:

Software Engineering module | PDF page 66 | lines 14-18

3.2.2. Non-Functional and Pseudo-Requirements
Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Nonfunctional requirements include quantitative constraints, such as response time (i.e., how fast the system reacts to user commands)
or accuracy (i.e., how precise are the system's numerical answers). The following are the

Question sources

Computer Science model exam 2 · Question 81

Question 84 | Q01261

Which of the following is false about functional requirement?

A. Requirements should complete

B. Requirement should be accurate as possible.

C. Each requirement should state a clear, independently verifiable obligation.

D. A requirement can be interpreted in many exclusive ways.

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer D: A requirement can be interpreted in many exclusive ways.

Requirements should accurately reflect stakeholder needs, cover the agreed scope and express clear, verifiable obligations. Unambiguity requires one intended interpretation; mutually exclusive readings violate that property. The revised third option states the useful single-obligation writing rule without claiming that duplicate requirements can never be consolidated. Thus the multiple-interpretation option is the single false quality claim, while completeness, accuracy and clarity remain valid goals.

Why each choice is right or wrong

A - Incorrect: Completeness requires coverage of the agreed behavior and exceptional cases.

B - Incorrect: Accuracy requires requirements to reflect the actual stakeholder needs.

C - Incorrect: A clear independently verifiable obligation supports precise specification.

D - Correct: Mutually exclusive readings violate the unambiguity requirement.

Primary note proof

Study-note section SE11

Correct requirements reflect actual stakeholder needs. Complete requirements address the agreed scope, normal and exceptional behavior. Consistency excludes contradictions. Clarity/unambiguity gives one intended interpretation; feasibility/realism means achievable

within constraints. Verifiability means objective evidence can check fulfillment. Replace “fast for most users” with a defined metric, workload and threshold.

Study-note section SE09

Write one clear obligation with an actor/trigger, action and observable result.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 68 | lines 18-26

system specification. Requirement validation involves checking if the specification is correct, complete, consistent, unambiguous, and realistic. A specification is correct if it represents the client’s view of the system (i.e., everything in the requirements model accurately represents an aspect of the system). It is complete if all possible scenarios through the system are described, including exceptional behavior (i.e., all aspects of the system are represented in the requirements
model). The system specification is consistent if it does not contradict itself. The system specification is unambiguous if exactly one system is defined (i.e., it is not possible to interpret the specification two or more different ways). Finally, it is realistic if the system can be implemented within constraints. These properties are illustrated with UML instance diagrams in

Question sources

Computer Science model exam 2 · Question 82

Question 85 | Q01272

Which of the following is not true about requirement elicitation?

A. Errors introduced during requirements elicitation are expensive to correct.

B. Define the purpose of the system by collaboration of the client and developers.

C. Requirements elicitation is less challenging than Analysis.

D. Requirements specification is the result of requirement elicitation.

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Requirements elicitation is less challenging than Analysis.

Elicitation requires stakeholder agreement and accurate problem understanding; analysis formalizes it. Elicitation is not inherently less challenging, and postponed decisions can be costly.

Why each choice is right or wrong

A - Incorrect: Late correction raises cost, consistent with identifying issues early. See PDF p. 62, 100.

B - Incorrect: Matches collaborative purpose definition. See PDF p. 62, 100.

C - Correct: The claimed easier-than-analysis ranking is unsupported and contrary to the described difficulty. See PDF p. 62, 100.

D - Incorrect: The system specification is its explicit output. See PDF p. 62, 100.

Primary note proof

Study-note section SE11

Correct requirements reflect actual stakeholder needs. Complete requirements address the agreed scope, normal and exceptional behavior. Consistency excludes contradictions. Clarity/unambiguity gives one intended interpretation; feasibility/realism means achievable within constraints. Verifiability means objective evidence can check fulfillment. Replace “fast for most users” with a defined metric, workload and threshold.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 62 | lines 9-16

Requirement’s elicitation focuses on describing the purpose of the system. The client, the developers, and the users identify a problem area and define a system that addresses the problem.
Such a definition is called a system specification and serves as a contract between the client and the developers. The system specification is structured and formalized during analysis to produce an analysis model (see Figure 3.1). Both system specification and analysis model represent the same information. They differ only in the language and notation they use. The system specification
is written in natural language, whereas the analysis model is usually expressed in a formal or semiformal notation.

Software Engineering module | PDF page 100 | lines 11-24

Analysis focuses on producing a model of the system, called the analysis model, which is correct, complete, consistent, and verifiable. Analysis is different from requirements elicitation in that developers focus on structuring and formalizing the requirements elicited from users
Figure 4.1 Products of requirements elicitation and analysis (UML activity diagram).
Althoughtheanalysismodelmaynotbeunderstandabletotheusersandtheclient, it helps developers verify the system specification produced during requirements elicitation. There is a natural tendency for users and developers to postpone difficult decisions until later in the project. A decision may be difficult because of lack of domain knowledge, lack of technological knowledge, or simply because of disagreements among users and developers. Postponing decisions enables the
project to move on smoothly and avoid confrontation with reality or peers. Unfortunately, difficult
decisions will eventually need to be made, often at higher cost when intrinsic problems are discovered during testing, or worse, during user evaluation. Translating a system specification in to a formal or semi-formal model forces developers to identify and resolve difficult issues early in
the development.

Question sources

Computer Science model exam 2 · Question 93

Question 86 | Q01512

Which named approach develops an experimental version specifically to explore unclear requirements or feasibility, and can therefore serve both as a development approach and an elicitation technique?

A. Spiral process model

B. Waterfall process model

C. Prototype process model

D. Agile process model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Prototype process model

Prototyping uses an early experimental representation or working version to discover requirements, usability needs or technical feasibility. It can be organized as a development approach and also used within elicitation. Spiral may contain a prototype to reduce risk, but risk-driven cycles define spiral itself. Waterfall names phased development, while Agile names a broader adaptive approach. The question asks the named experimentation technique that directly serves both purposes.

Why each choice is right or wrong

A - Incorrect: Spiral may employ prototypes but is itself defined by risk-driven cycles.

B - Incorrect: Waterfall is a phased lifecycle rather than the experimentation technique described.

C - Correct: Prototyping directly explores requirements or feasibility using an experimental version.

D - Incorrect: Agile is a broad adaptive approach and can use several elicitation techniques.

Primary note proof

Study-note section SE07

> A prototype explores requirements, usability or technical feasibility. A throwaway prototype is discarded after learning; an evolutionary prototype is deliberately refined toward production. A quick prototype may omit robustness, security or architecture, so do not deploy it unchanged without evaluation. Coding may occur during prototyping even if a diagram labels that activity construction rather than coding.

Study-note section SE06

Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 26 | lines 6-9

2. Requirement's elicitation and analysis: This is the process of deriving the system requirements through observation of existing systems, discussions with potential users and procurers, task analysis, and so on. This may involve the development of one or more system models and prototypes. These help you understand the system to be specified.

Software Engineering module | PDF page 82 | lines 23-29

• Prototype test. During this type of test, one or more users are presented with a piece of software that practically implements the system. A vertical prototype implements a complete use case through the system, and a horizontal prototype presents an interface for most use cases (without providing much or any functionality). The advantages of prototype tests are that they provide a realistic view of the system to the user and that prototypes can be instrumented to collect detailed data. The disadvantages of prototypes are that they are expensive to build and to modify.

Question sources

Mock exam 1 · Question 28

Question 87 | Q01513

What is the base for functional testing?

A. We have to refer to the system design document

B. We have to refer to the system requirement specification document

C. We have to refer to user manual if any exists

D. A and B

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: We have to refer to the system requirement specification document

Functional testing tests requirements from the RAD and, if available, the user manual. Of the options, the requirements specification is the primary basis; the user manual is supplementary.

Why each choice is right or wrong

A - Incorrect: Design goals support performance tests and architecture supports integration, rather than being the primary functional oracle. See PDF p. 144, 83, 130.

B - Correct: The requirement document defines what behavior to test. See PDF p. 144, 83, 130.

C - Incorrect: A manual can supplement functional requirements, but is not the primary complete basis. See PDF p. 144, 83, 130.

D - Incorrect: Adding design as an equal required functional basis overstates the cited rule. See PDF p. 144, 83, 130.

Primary note proof

Study-note section SE36

Black-box techniques derive cases from specified behavior; white-box techniques derive them from internal structure; grey-box uses partial internal knowledge. Regression testing checks for unintended effects after change; confirmation testing checks the particular defect repair. Exploratory testing combines learning, design and execution and is experience based. Manual/automated describe execution support, not a compulsory black-/white-box classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Software Engineering module | PDF page 83 | lines 12-25

3.4.4. Documenting Requirements Elicitation
The results of the requirements elicitation activity and the analysis activity are documented in the
Requirements Analysis Document (RAD). This document completely describes the system in terms of functional and nonfunctional requirements and serves as a contractual basis between the

client and the developers. The audience for the RAD includes the client, the users, the project management, the system analysts (i.e., the developers who participate in the requirements), and the system designers (i.e., the developers who participate in the system design). The first part of the document, including use cases and nonfunctional requirements, is written during requirements
elicitation. The formalization of the specification in terms of object models is written during analysis.
The first section of the RAD is an Introduction. Its purpose is to provide a brief overview of the function of the system and the reasons for its development, its scope, and references to the development context (e.g., related problem statement, references to existing systems, feasibility studies). The introduction also includes the objectives and success criteria of the project.

Software Engineering module | PDF page 130 | lines 3-11

6.3. DOCUMENTING SYSTEM DESIGN
System design is documented in the System Design Document (SDD). It describes design goals set by the project, the subsystem decomposition (with UML class diagrams), the hardware/software mapping (with UML deployment diagrams), the data management, the access control, control flow mechanisms, and boundary conditions. The SDD is used to define interfaces between teams of developers and as reference when architecture-level decisions need to be revisited. The audience for the SDD includes the project management, the system architects (i.e., the developers who participate in the system design), and the developers who design and implement each subsystem. The following is an example template for a SDD:

Question sources

Mock exam 1 · Question 29

Question 88 | Q01514

Evaluate statements A-C about subsystem decomposition. Which is false? Select D if none is false.

A. In designing the desirable effect is low coupling and high cohesion in sub system decomposition

B. Cohesion refers to the dependency of classes with in subsystem

C. Coupling refers to the dependency between sub systems

D. None of statements A-C is false.

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: None of statements A-C is false.

Low coupling limits dependencies between subsystems, while high cohesion keeps related responsibilities together inside each subsystem. The first statement recommends this combination, the second identifies cohesion within the subsystem, and the third identifies coupling across subsystems. All three substantive statements are therefore true. The final alternative is explicitly the response for finding no false statement among those three, rather than a self-referential assertion about all choices.

Why each choice is right or wrong

A - Incorrect: Low coupling and high cohesion are the stated decomposition goals.

B - Incorrect: Cohesion concerns relatedness and dependencies among elements within a subsystem.

C - Incorrect: Coupling concerns dependencies between subsystems.

D - Correct: None of the substantive statements A-C is false, so this is the required response.

Primary note proof

Study-note section SE22

> Cohesion measures how well elements inside a module serve a related purpose; coupling measures dependency between modules. Seek high cohesion and low coupling. Shared mutable data and knowledge of another module’s internal representation increase coupling. Information hiding exposes stable operations while concealing changeable decisions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 120 | lines 6-12

> 6.1.3. Coupling and coherence
> Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
> is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Software Engineering module | PDF page 120 | lines 24-27

> Coherence is the strength of dependencies within a subsystem. If a subsystem contains many objects that are related to each other and perform similar tasks, its coherence is high. If a subsystem
> contains a number of unrelated objects, its coherence is low. A desirable property of a subsystem decomposition is that it leads to subsystems with high coherence.

Question sources

Mock exam 1 · Question 30

Question 89 | Q01593

Which offered asset is inherently excluded from reuse in a software product line?

A. Test cases

B. Requirement

C. Architecture

D. None of A-C; all three can be reusable product-line assets.

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer D: None of A-C; all three can be reusable product-line assets.

Product-line assets can include requirements, architecture, components and tests. Consequently test cases, requirements and architecture are all potentially reusable, although each must be evaluated for fit in a particular product. None of A-C is inherently excluded by its asset type. Option D expresses this conclusion directly rather than contradicting the negative stem with an undefined all-choices assertion.

Why each choice is right or wrong

A - Incorrect: Test cases can be reusable product-line assets.

B - Incorrect: Requirements can be shared across related products.

C - Incorrect: Architecture can be a reusable product-line asset.

D - Correct: None of A-C is inherently excluded; all can be reused.

Primary note proof

Study-note section SE07

> RAD uses rapid construction, timeboxing, user participation and often reusable components; it requires available users and skilled designers/developers. Reuse-oriented development searches components, evaluates fit, may adjust requirements, designs around interfaces and builds missing parts. Product-line assets may include requirements, architecture, components and tests; all are potentially reusable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 24 | lines 20-29

> Advantages:
> 1. Reduced cost and risks
> 2. Faster delivery of the software (Accelerated development)
> Disadvantages:
> 1. Compromising requirements

2. Some control over the system evolution is lost as new versions of the reusable components are not under the control of the organization using them.
3. Increased maintenance costs
4. Not-invented-here syndrome
5. Lack of tool support

Question sources

Mock exam 3 · Question 25

Question 90 | Q01594

A software design model states exact withdraw preconditions and balance equations but omits physical implementation detail. Compare it with an informal mathematical sketch that leaves those constraints unspecified and a fully detailed engine drawing. Which statement is FALSE for these particular models?

A. This software model is less precise than the specified mathematical sketch.

B. This software model is more abstract than the fully detailed engine drawing.

C. A main role of this model is to represent the software designer’s ideas.

D. None of A-C is false.

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer A: This software model is less precise than the specified mathematical sketch.

The software model states exact preconditions and balance equations, whereas the informal mathematical sketch omits those constraints. It is therefore not less precise in the stated comparison, making A false. Omitting implementation detail makes it more abstract than the specified detailed engine drawing, and representing design ideas remains a legitimate purpose. These are scenario-specific judgments rather than universal rankings of all software and mathematical models.

Why each choice is right or wrong

A - Correct: Its exact constraints make the stated less-precise comparison false.

B - Incorrect: The software model omits implementation detail retained in the detailed drawing.

C - Incorrect: A design model can represent the designer’s ideas and decisions.

D - Incorrect: A is false, so the no-false-statements alternative is wrong.

Primary note proof

Study-note section SE43

Choose designs by their effect on stated goals. Fine-grained self-contained components, separation of producers/consumers and limited shared mutable data often improve maintainability. Trade-offs remain: interfaces can add latency, isolation can complicate transactions, and global state can simplify one operation while coupling many others. Prefer measurable quality scenarios to counting components alone.

Study-note section SE25

A contract states preconditions, postconditions and invariants. For withdraw(amount): require amount>0 and amount<=balance; ensure balance=oldBalance-amount; preserve balance>=0. Validation checks inputs; implementation enforces the transition. A complete feature needs both UI/API and persisted behavior, not only a diagram.

Study-note section SE23

A layer is a logical abstraction level; a tier is a deployment separation. A closed layered architecture permits calls only to the immediately lower layer; an open one permits approved skipping. For top-to-bottom A,B,C,D, A→B obeys closed layering, A→C skips, and lower-to-upper calls violate the simple dependency rule unless explicitly mediated by a defined mechanism.

Study-note section SE23

A structure is the elements/relations themselves; a view represents selected aspects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 100 | lines 11-24

Analysis focuses on producing a model of the system, called the analysis model, which is correct, complete, consistent, and verifiable. Analysis is different from requirements elicitation in that developers focus on structuring and formalizing the requirements elicited from users
Figure 4.1 Products of requirements elicitation and analysis (UML activity diagram).
Althoughtheanalysismodelmaynotbeunderstandabletotheusersandtheclient, it helps developers verify the system specification produced during requirements elicitation. There is a natural tendency for users and developers to postpone difficult decisions until later in the project. A decision may be difficult because of lack of domain knowledge, lack of technological knowledge, or simply because of disagreements among users and developers. Postponing decisions enables the
project to move on smoothly and avoid confrontation with reality or peers. Unfortunately, difficult
decisions will eventually need to be made, often at higher cost when intrinsic problems are discovered during testing, or worse, during user evaluation. Translating a system specification in to a formal or semi-formal model forces developers to identify and resolve difficult issues early in
the development.

Software Engineering module | PDF page 18 | lines 13-20

System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
Software design involves identifying and describing the fundamental software system

abstractions
and their relationships. The goal of the design phase is to transform the requirements specified in
the SRS document into a structure that is suitable for implementation in some programming language. In technical terms, during the design phase the software architecture is derived from the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Question sources

Mock exam 3 · Question 26

Question 91 | Q01595

Which among the following designing models show sequence functional transformation done on inputs into outputs?

A. ER diagrams

B. DFD diagrams

C. Component diagrams

D. None of the choices are correct

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: Data-flow diagrams.

SE20 defines a DFD through data flows, stores and transformations from inputs to outputs. That matches the functional-transformation focus in the stem. An ER diagram focuses on domain data relationships, while a component diagram describes software components and dependencies. A DFD need not show detailed control timing merely because it depicts connected transformations.

Why each choice is right or wrong

A - Incorrect: ER diagrams describe entities and relationships rather than transformations of input data into outputs.

B - Correct: DFDs depict data flows and the processes that transform inputs into outputs.

C - Incorrect: Component diagrams describe component interfaces and dependencies rather than this data-transformation view.

D - Incorrect: DFD is listed and directly matches the functional-transformation description.

Primary note proof

Study-note section SE20

Choose a sequence diagram for message order between participants, a state machine for event-dependent state changes, and activity for workflow. Both sequence and state diagrams describe behavior over time. State machines are useful for objects with significant states, whether entity or control. A DFD shows data stores/flows and transformations from inputs to outputs; it is not a UML diagram.

Study-note section SE-S03

An entity-relationship diagram models entity types, their attributes and the relationships between them, commonly for data design. Its focus differs from a use-case diagram of external roles and goals, a class diagram of classes and their features, or a deployment diagram mapping artifacts to execution nodes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 126 | lines 7-16

Pipe and filter architecture
In the pipe and filter architecture (see Figure 6.13), subsystems process data received from a set of
inputs and send results to other subsystems via a set of outputs. The subsystems are called filters, and the associations between the subsystems are called pipes. Each filter only knows the content and the format of the data received on the input pipes, not the filters that produced them. Each filter is executed concurrently and synchronization is done via the pipes. The pipe and filter architecture is modifiable: Filters can be substituted for others or reconfigured to achieve a different purpose.
Figure 6.13 Pipe and filter architecture (UML class diagram). A Filter can have many inputs and outputs. A Pipe connects one of the outputs of a Filter to one of the inputs of another Filter.

Software Engineering module | PDF page 101 | lines 5-6

In UML, the functional model is represented with use case diagrams, the object model with class diagrams, and the dynamic model with statechart and sequence diagrams.

Question sources

Mock exam 3 · Question 27

Question 92 | Q01596

Which of the following is step in software development life cycle that tell us if developing a software makes sense for the current company situation or not.

A. Requirement analysis

B. Feasibility analysis

C. Designing phase

D. Requirement elicitation

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: Feasibility analysis

Feasibility assesses technological practicality, budget and business cost-effectiveness before deciding to proceed with detailed analysis.

Why each choice is right or wrong

A - Incorrect: Further analysis follows a feasibility decision. See PDF p. 25, 26.

B - Correct: Direct viability/go-ahead decision. See PDF p. 25, 26.

C - Incorrect: Derives a solution after requirements are understood. See PDF p. 25, 26.

D - Incorrect: Collects stakeholder needs rather than deciding whether the investment is viable. See PDF p. 25, 26.

Primary note proof

Study-note section SE08

> Feasibility asks whether a proposal is worth doing and can be delivered within technical, economic, operational and schedule constraints. Examine existing limitations, organizational impacts, benefits, cost, available skills and integration risk. System analysis studies the problem and needed behavior in more detail; the activities inform each other.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 25 | lines 15-18

> There are four main activities in the requirements engineering process:
> 1. Feasibility study: Feasibility shows viability and practicality of a software, project or idea. An estimate is made of whether the identified user needs may be satisfied using current software and hardware technologies. The study considers whether the proposed system will be cost-

Software Engineering module | PDF page 26 | lines 3-5

> effective from a business point of view and if it can be developed within existing budgetary constraints. A feasibility study should be relatively cheap and quick. The result should inform the decision of whether or not to go ahead with a more detailed analysis.

Question sources

Mock exam 3 · Question 28

Question 93 | Q01597

Which choice gives the most complete supported comparison of object-oriented model continuity and transitions between different development models?

A. Object-oriented development can use a consistent vocabulary of objects and classes across analysis and design.

B. Reuse and consistent models in object-oriented development can reduce development effort and duration on suitable projects.

C. Moving between different representations in a traditional phased approach can add translation complexity and errors.

D. All statements A-C are correct.

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: All statements A-C are correct.

A consistent object vocabulary across development stages can reduce errors from translating between unrelated models. Reuse and continuity can also reduce effort on suitable projects, while transitions between different representations can add complexity. These are potential benefits and risks, not guarantees that every OO project is shorter or every traditional project is worse. With those qualifications, all three substantive statements are supported, making their aggregate the most complete answer.

Why each choice is right or wrong

A - Incorrect: True but incomplete alone: a consistent object/class vocabulary can connect analysis and design.

B - Incorrect: True but incomplete alone: reuse and model continuity can reduce effort, without guaranteeing it.

C - Incorrect: True but incomplete alone: translation between different representations can add complexity or errors.

D - Correct: All three qualified statements hold, so the aggregate is the complete selection.

Primary note proof

Study-note section SE01

> Software engineering applies systematic methods to development, operation and maintenance. A software product includes executable code and relevant requirements, designs, test assets, configuration and user documentation. A process defines activities, ordering, roles, products and entry/exit conditions. Its four broad activities are specification, design/implementation, validation and evolution.

Study-note section SE01

Procedural decomposition organizes functions; object oriented decomposition organizes objects with state and behavior. Either can use iterative or sequential development. OOP does not automatically shorten every project. Reuse, encapsulation and consistent models can reduce duplication and ease change; poor object design can still be complex. Functional programming emphasizes function evaluation and controlled side effects; event-driven programs react to events.

Study-note section SE-S02

An object-oriented approach can use a consistent vocabulary of objects, classes and relationships across analysis, design and implementation. Models are refined for different purposes rather than being identical artifacts. This continuity can reduce translation errors and effort; reusable classes can shorten suitable projects, but neither shorter duration nor good design is guaranteed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 31 | lines 19-31

1.3.1. Why an object oriented
• Object oriented systems are
o Easier to adapt to changes
o More robust
o Easier to maintain
o Promote greater design and code reuse
o Creates modules of functionality
• REASONS FOR WORKING OF OBJECT ORIENTED SYSTEMS:
Higher level of abstraction
The object-oriented approach supports abstraction at the object level. Since objects encapsulate both data (attributes) and functions (methods), they work at a higher level of abstraction. The development can proceed at the object level and ignore the rest of the system for as long as necessary. This makes designing, coding, testing, and maintaining the system much simpler.

Software Engineering module | PDF page 32 | lines 22-27

1.3.2. Overview of the unified approach.
The unified modeling language (UML) is a set of notations and conventions used to describe and model an application. But, the UML does not specify a methodology or what steps to follow to develop an application; that would be the task of the UA. Figure 1.2 depicts the essence of the unified approach. The heart of the UA is Jacobson's use case. The use case represents a typical interaction between a user and a computer system to capture the users' goals and needs. The main

Software Engineering module | PDF page 12 | lines 3-17

Table 1.1. : Comparison of Traditional Approach vs Object oriented system
TRADITIONAL APPROACH OBJECT ORIENTED SYSTEM
DEVELOPMENT
Collection of procedures(functions) Combination of data and functionality
Focuses on function and procedures,
different styles and methodologies for each

step of process
Focuses on object, classes, modules that
can be easily replaced, modified and reused.
Moving from one phase to another phase
is complex.
Moving from one phase to another phase
is easier.
Increases duration of project decreases duration of project
Increases complexity Reduces complexity and redundancy

Software Engineering module | PDF page 32 | lines 3-10

The traditional approach to software development requires different styles and methodologies for
each step of the process. Moving from one phase to another requires a complex transition of perspective between models that almost can be in different worlds. This transition not only can slow the development process but also increases the size of the project and the chance for errors introduced in moving from one language to another. The object-oriented approach, on the other hand, essentially uses the same language to talk about analysis, design, programming, and database
design. This seamless approach reduces the level of complexity and redundancy and makes for clearer, more robust system development.

Question sources

Mock exam 3 · Question 29

Question 94 | Q01598

Which of the following process model is concerned with how changes are incorporated in software?

A. Configuration management process

B. Change management process

C. Inspection management process

D. Project management process

Hint

Separate artifact history and controlled changes from scheduling, collaboration benefits and automated execution.

Answer B: Change management process.

SE33 assigns handling change requests, impact assessment, decisions, implementation and validation to change management. Configuration management coordinates versions and baselines of affected artifacts, which supports that work but is a different focus. Inspection and general project management also contribute, yet B specifically names how proposed software changes are evaluated and incorporated.

Why each choice is right or wrong

A - Incorrect: Configuration management maintains controlled versions and baselines rather than primarily deciding and incorporating requested changes.

B - Correct: Change management evaluates, approves/prioritizes, implements and validates proposed changes.

C - Incorrect: Inspection evaluates artifacts rather than defining the complete change-handling process.

D - Incorrect: Project management is broader than the specific process for handling software change requests.

Primary note proof

Study-note section SE33

> Change management handles request, impact assessment, decision/prioritization, implementation and validation; configuration management keeps all affected versions consistent. Distributed Git gives each clone local history and supports independent branches followed by merging; it does not eliminate merge conflicts or approvals. Plan test configuration management early and apply it throughout testing, not at just one exclusive phase.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 142 | lines 5-10

> Configuration management avoids faults caused by undisciplined change in the system models. For example, it is a common mistake to a change a subsystem interface without notifying all developers of calling components. Configuration management can make sure that, if analysis models and code are becoming inconsistent with one another, analysts and implementor are notified. The assumption behind configuration management is that the system contains many less
> faults if change is controlled.

Question sources

Mock exam 3 · Question 30

Question 95 | Q01599

After a maintenance change request has been received, which is the earliest listed technical activity before release planning and implementation?

A. Release planning

B. Change management

C. Impact analysis

D. Testing

Hint

Classify the motive for the change before its implementation: repair, environment, useful capability or reduced future risk.

Answer C: Impact analysis

A typical evolution sequence is change request, impact analysis, release planning, implementation/testing and release. Once the request has arrived, impact analysis is the earliest listed technical step, assessing consequences before scheduling and implementation. Change management names the broader handling process rather than that next step. Release planning and testing occur later, so neither starts the specified post-request technical sequence.

Why each choice is right or wrong

A - Incorrect: Release planning follows impact analysis.

B - Incorrect: Change management is the overarching process.

C - Correct: Impact analysis is the first listed technical activity after the request.

D - Incorrect: Testing follows analysis, planning and implementation.

Primary note proof

Study-note section SE40

> A typical evolution sequence is change request→impact analysis→release planning→implementation/testing→release. Among a list lacking the request step, impact analysis may be the earliest listed technical activity. Major restructuring is reengineering; if the architecture itself changes, architectural transformation is the more specific description. Markets, technology, stakeholders and funding can all change scope or priorities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 19 | lines 24-29

> 5. Operation and maintenance: Normally (although not necessarily), this is the longest life cycle phase. Maintenance of a typical software product requires much more than the effort necessary to
> develop the product itself. The system is installed and put into practical use. Maintenance involves

correcting errors which were not discovered in earlier stages of the life cycle, improving the implementation of system units and enhancing the system ‘s services as new requirements are discovered.

Software Engineering module | PDF page 41 | lines 16-17

✓ Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous and the software must evolve to remain useful.

Question sources

Mock exam 3 · Question 31

Question 96 | Q01600

A takes 5 days. B takes 3 days and C takes 4 days; both depend only on A and may run in parallel with sufficient resources. The project finishes when all three complete. What is its earliest duration?

A. 3 days

B. 4 days

C. 5 days

D. 9 days

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer D: 9 days

A completes at day five. B and C then start together, finishing at days eight and nine respectively. The project must wait for both required tasks, so its earliest finish is 5+max(3,4)=9 days. Adding all durations would ignore the permitted parallelism, while using eight days would count only the shorter A-B path and leave C incomplete.

Why each choice is right or wrong

A - Incorrect: Three counts B alone and omits its predecessor.

B - Incorrect: Four counts C alone and omits A.

C - Incorrect: Five stops after A, before both successors.

D - Correct: A followed by the longer parallel branch takes nine days.

Primary note proof

Study-note section SE29

If A takes 5 days and B=3,C=4 both follow A, project duration is 5+max(3,4)=9 days. B finishes at 8, but the project waits for C at 9. If paths B5+C6 and D8+E3 share a zero-duration start/end, both total 11; there are two critical paths, not one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 3-12

4.6. Scheduling, Organization and Team Structures
Project-task scheduling is an important project planning activity. It involves deciding which tasks would
be taken up when. In order to schedule the project activities, a software project manager needs to do
the following:
• Identify all the tasks needed to complete the project and Break down large tasks into small activities.
• Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
• Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Mock exam 3 · Question 32

Question 97 | Q01601

A project starts at day 0 and must finish at day 9. It consists of exactly two parallel paths, A → B and C → D. A and C have no predecessors, B depends only on A, D depends only on C, and the project finishes when both B and D are complete. Activity durations and early times are: A: duration 4 days, ES=0, EF=4; B: duration 5 days, ES=4, EF=9; C: duration 3 days, ES=0, EF=3; D: duration 6 days, ES=3, EF=9. What is the total slack (float) of Activity C?

A. 0 days

B. 1 day

C. 3 days

D. 6 days

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer A: 0 days.

Each complete path takes 9 days: A→B is 4+5 and C→D is 3+6. Working backward from day 9, D must start by day 3. Therefore C must finish by day 3 and start by day 0. C has LS=0 and ES=0, so total slack = LS−ES = 0 days (equivalently LF−EF = 3−3). A is correct. Delaying C would delay its successor D and the required project finish.

Why each choice is right or wrong

A - Correct: C has latest start 0 and earliest start 0, giving zero total slack.

B - Incorrect: A one-day delay would make the C→D path finish on day 10, beyond the day-9 finish.

C - Incorrect: C's three-day duration is not its slack; a three-day delay would finish this path on day 12.

D - Incorrect: D's six-day duration is not C's slack; a six-day delay to C would finish the path on day 15.

Primary note proof

Study-note section SE30

> Early start/finish values alone do not determine slack without successor relationships and the project completion constraint. If C3→D6 is a complete path parallel to A4→B5 and both finish at day9, C has zero float. That network cannot be silently inferred from four unrelated ES/EF rows.

Study-note section SE30

> Traditional CPM uses point durations; PERT models uncertainty with optimistic a, most-likely m and pessimistic b. The classic estimate is TE=(a+4m+b)/6 and variance=((b-a)/6)^2. For 3,6,15 days, TE=42/6=7 days and variance=4 days². Do not replace the weighted estimate by a simple average. These formulas are approximations, not guaranteed actual completion times.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 3-12

> 4.6. Scheduling, Organization and Team Structures
> Project-task scheduling is an important project planning activity. It involves deciding which tasks would
> be taken up when. In order to schedule the project activities, a software project manager needs to do
> the following:
> • Identify all the tasks needed to complete the project and Break down large tasks into small activities.
> • Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
> • Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Mock exam 3 · Question 33

Question 98 | Q01602

PERT considers which of the following time estimates for each activity?

A. Ideal time only

B. Worst-case time only

C. Three-point estimates (optimistic, most likely, pessimistic)

D. Average of historical data (if available) statement.

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer C: Optimistic, most likely and pessimistic estimates.

SE30 defines the classic PERT inputs as optimistic a, most-likely m and pessimistic b. These feed the weighted expected time (a+4m+b)/6 and an uncertainty estimate. Neither a single ideal or worst-case time nor an unqualified historical average supplies the three-point representation of uncertainty described by PERT.

Why each choice is right or wrong

A - Incorrect: An ideal estimate alone omits the most-likely and pessimistic values needed by classic PERT.

B - Incorrect: A worst-case estimate alone omits the optimistic and most-likely possibilities.

C - Correct: Classic PERT uses optimistic, most-likely and pessimistic duration estimates.

D - Incorrect: Historical data may inform estimates, but an average alone is not the PERT three-point input model.

Primary note proof

Study-note section SE30

> Traditional CPM uses point durations; PERT models uncertainty with optimistic a, most-likely m and pessimistic b. The classic estimate is TE=(a+4m+b)/6 and variance=((b-a)/6)^2. For 3,6,15 days, TE=42/6=7 days and variance=4 days^2. Do not replace the weighted estimate by a simple average. These formulas are approximations, not guaranteed actual completion times.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 3-12

> 4.6. Scheduling, Organization and Team Structures
> Project-task scheduling is an important project planning activity. It involves deciding which tasks

would
be taken up when. In order to schedule the project activities, a software project manager needs to do
the following:
• Identify all the tasks needed to complete the project and Break down large tasks into small activities.
• Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
• Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Mock exam 3 · Question 34

Question 99 | Q01603

_____diagram is not supported by UML considering Data-driven modeling.

A. Activity

B. Data Flow Diagram (DFD)

C. State Chart

D. Component

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: Data Flow Diagram (DFD)

The traditional nine UML diagram types include activity, statechart and component diagrams; DFD is not a UML diagram type. Thus DFD is the offered non-UML notation.

Why each choice is right or wrong

A - Incorrect: Listed UML activity diagram. See PDF p. 101, 56, 55.

B - Correct: Not included among the module's nine UML diagrams on PDF p.47. See PDF p. 101, 56, 55.

C - Incorrect: Listed UML statechart diagram. See PDF p. 101, 56, 55.

D - Incorrect: Listed UML component diagram on PDF p.47. See PDF p. 101, 56, 55.

Primary note proof

Study-note section SE20

Choose a sequence diagram for message order between participants, a state machine for event-dependent state changes, and activity for workflow. Both sequence and state diagrams describe behavior over time. State machines are useful for objects with significant states,

whether entity or control. A DFD shows data stores/flows and transformations from inputs to outputs; it is not a UML diagram.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 101 | lines 5-6

In UML, the functional model is represented with use case diagrams, the object model with class diagrams, and the dynamic model with statechart and sequence diagrams.

Software Engineering module | PDF page 56 | lines 18-24

2.3.4. Activity diagrams
The outgoing transitions are triggered by the completion of an action associated with the state. This is called an action state. By convention, the name of a state denotes a condition, whereas the name of an action state denotes an action. Activity diagrams are statechart diagrams whose states
are action states. is an activity diagram corresponding to the state diagram. An alternate and equivalent view of activity diagrams is to interpret action states as control flow between activities
and transitions; that is, the arrows are interpreted as sequential constraints between activities.

Software Engineering module | PDF page 55 | lines 4-9

2.3.3. State chart Diagram
A UML statechart is a notation for describing the sequence of states an object goes through in response to external events. Statecharts are extensions of the finite state machine model. On the one hand, statecharts provide notation for nesting states and state machines (i.e., a state can be described by a state machine). On the other hand, statecharts provide notation for binding transitions with message sends and conditions on objects.

Software Engineering module | PDF page 47 | lines 15-27

UML includes the following nine diagrams and the details are described in the following chapters.
• Class diagram
• Object diagram
• Use case diagram
• Sequence diagram
• Collaboration
diagram
• Activity diagram
• State chart diagram
• Deployment
diagram
• Component
diagram

Question sources

Mock exam 3 · Question 35

Question 100 | Q01604

What defining intention distinguishes a throwaway prototype from an evolutionary prototype?

A. It involves successive steps

B. It involves just one task

C. The prototype is built with the idea that it will eventually be converted into final system

D. It is intended to be discarded after learning, rather than refined into the production system.

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer D: It is intended to be discarded after learning, rather than refined into the production system.

A throwaway prototype is built to learn about requirements, usability or feasibility and is discarded after serving that purpose. An evolutionary prototype is deliberately refined toward production. Either may involve successive steps, multiple tasks and varying development time, so those features do not define the difference. The artifact's intended future, rather than how quickly it was built, distinguishes the approaches.

Why each choice is right or wrong

A - Incorrect: Either kind may involve successive steps.

B - Incorrect: Either kind can explore more than one task.

C - Incorrect: Intended refinement into the final system describes evolutionary prototyping.

D - Correct: Discard after learning is the defining throwaway intention.

Primary note proof

Study-note section SE07

> A prototype explores requirements, usability or technical feasibility. A throwaway prototype is discarded after learning; an evolutionary prototype is deliberately refined toward production. A quick prototype may omit robustness, security or architecture, so do not deploy it unchanged without evaluation. Coding may occur during prototyping even if a diagram labels that activity construction rather than coding.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 82 | lines 23-29

> • Prototype test. During this type of test, one or more users are presented with a piece of software that practically implements the system. A vertical prototype implements a complete use case through the system, and a horizontal prototype presents an interface for most use cases (without providing much or any functionality). The advantages of prototype

tests are that they provide a realistic view of the system to the user and that prototypes can be instrumented to collect detailed data. The disadvantages of prototypes are that they are expensive to build and to modify.

Software Engineering module | PDF page 26 | lines 6-9

2. Requirement's elicitation and analysis: This is the process of deriving the system requirements through observation of existing systems, discussions with potential users and procurers, task analysis, and so on. This may involve the development of one or more system models and prototypes. These help you understand the system to be specified.

Question sources

Mock exam 3 · Question 36

Question 101 | Q01605

Which factors can cause requirements, scope or priorities to evolve during a project? Select all that apply.

A. Changing needs of stakeholders

B. Advances in technology

C. Market dynamics

D. Lack of project funding

Hint

Classify the motive for the change before its implementation: repair, environment, useful capability or reduced future risk.

Answer A, B, C, D: Changing needs of stakeholders; Advances in technology; Market dynamics; Lack of project funding

Stakeholder needs can change the services required, technology can introduce new constraints or opportunities, and market dynamics can alter business priorities. A funding shortfall can also reduce scope or force reprioritization. The note explicitly includes all four sources of change. Their effects depend on the project, but each can cause evolution, so none is an unconditional exception.

Why each choice is right or wrong

A - Correct: Changed stakeholder needs can revise required behavior.

B - Correct: Technology changes can alter constraints or opportunities.

C - Correct: Market changes can alter goals and priorities.

D - Correct: Funding limits can reduce scope or reorder requirements.

Primary note proof

Study-note section SE11

Trace each requirement backward to its source/rationale and forward to design, code and tests. Record dependencies among requirements. A traceability matrix links these artifacts and helps impact analysis; it is not just a cost/schedule ledger. Stakeholders must negotiate conflicting requirements rather than independently approving incompatible demands. Scope, understanding and volatility are distinct elicitation difficulties.

Study-note section SE40

Corrective maintenance fixes faults; adaptive maintenance responds to environment changes; perfective maintenance improves useful capability/performance; preventive maintenance reduces future risk, such as restructuring fragile code. Motive determines category; refactoring can support preventive or perfective work. Maintenance often lasts years and includes understanding legacy code and regression effort, so it may dominate lifecycle cost.

Study-note section SE40

Markets, technology, stakeholders and funding can all change scope or priorities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 41 | lines 16-17

✓ Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous and the software must evolve to remain useful.

Software Engineering module | PDF page 96 | lines 12-21

4.7. Risk Management
• Risk management is concerned with identifying risks and drawing up plans to minimise their effect on a project.
• A risk is a probability that some adverse circumstance will occur
- Project risks affect schedule or resources;
- Product risks affect the quality or performance of the software being developed;
- Business risks affect the organisation developing or procuring the software.
Risk Management Process
- Risk identification- Identify project, product and business risks;
- Risk analysis -Assess the likelihood and consequences of these risks;

Question sources

Mock exam 3 · Question 37

Question 102 | Q01606

Used when a large amount of testing is needed and the changes, while small, can affect many parts of the system

A. Preventive testing

B. Unit Testing

C. Regression Testing

D. Acceptance Testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer C: Regression testing.

The concern is that small changes may disturb other previously working parts of the software. Regression testing means checking unintended effects after change, rather than only confirming the defect. Such testing can span multiple levels and a large scope; the question describes its purpose, not the size of a unit or an acceptance decision.

Why each choice is right or wrong

A - Incorrect: Preventive testing is not the defined technique for detecting unintended effects of a specific software change.

B - Incorrect: Unit testing isolates a small component and does not by itself define the cross-system change-impact purpose.

C - Correct: Regression tests check whether changes unintentionally damage previously working behavior elsewhere.

D - Incorrect: Acceptance testing supports a user acceptance decision rather than specifically checking unintended change effects.

Primary note proof

Study-note section SE36

> Black-box techniques derive cases from specified behavior; white-box techniques derive them from internal structure; grey-box uses partial internal knowledge. Regression testing checks for unintended effects after change; confirmation testing checks the particular defect repair. Exploratory testing combines learning, design and execution and is experience based. Manual/automated describe execution support, not a compulsory black-/white-box classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 142 | lines 5-10

> Configuration management avoids faults caused by undisciplined change in the system models. For example, it is a common mistake to a change a subsystem interface without notifying all

developers of calling components. Configuration management can make sure that, if analysis models and code are becoming inconsistent with one another, analysts and implementor are notified. The assumption behind configuration management is that the system contains many less
faults if change is controlled.

Software Engineering module | PDF page 147 | lines 14-18

• Unit testing, which finds faults by isolating an individual component using test stubs and drivers and by exercising the component using a test case
• Integration testing, which finds faults by integrating several components together
• System testing, which focuses on the complete system, its functional and nonfunctional requirements, and its target environment

Software Engineering module | PDF page 147 | lines 3-8

• A test stub is a partial implementation of components on which the tested component depends. A test driver is a partial implementation of a component that depends on the tested component. Test stubs and drivers enable components to be isolated from the rest of the system for testing.
• A correction is a change to a component. The purpose of a correction is to repair a fault. Note that a correction can introduce new faults.

Question sources

Mock exam 3 · Question 38

Question 103 | Q01607

The Decision table testing is a ----------?

A. White box Test Design Technique

B. Black Box Test Design Technique

C. Experience-based Test Design Technique

D. Grey Box Test Design Technique

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: Black-box, specification-based testing.

SE37 derives decision-table cases from combinations of specified conditions and resulting actions. This is specification-based rather than code-structure-based testing. It therefore belongs to black-box techniques. Experience or partial internal knowledge may help a tester, but they do not define the decision-table method or change its condition/action test basis.

Why each choice is right or wrong

A - Incorrect: White-box techniques derive tests from internal implementation structure rather than specified condition/action rules.

B - Correct: Decision tables derive cases from specified conditions and expected actions, making them black-box.

C - Incorrect: The technique has an explicit specification-based structure rather than being defined by tester experience.

D - Incorrect: Partial knowledge of internals is not required to derive decision-table test cases.

Primary note proof

Study-note section SE37

> A decision table lists condition combinations and resulting actions. For validCard and enoughFunds, enumerate TT,TF,FT,FF; specify dispense only for TT and the proper rejection for others. Cover feasible rules rather than blindly using every impossible combination. These are specification-based techniques. Test requirements can also define state-transition cases, such as rejecting dispatch from an already closed incident.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 149 | lines 6-10

> Equivalence testing is a black box testing technique that minimizes the number of test cases. The possible inputs are partitioned into equivalence classes, and a test case is selected for each class. The assumption of equivalence testing is that systems usually behave in similar ways for all members of a class. To test the behavior associated with an equivalence class, we only need to test
> one member of the class. Equivalence testing consists of two steps: identification of the

Software Engineering module | PDF page 149 | lines 24-29

> Path testing
> Path testing is a white box testing technique that identifies faults in the implementation of the component. The assumption behind path testing is that, by exercising all possible paths through the code at least once, most faults will trigger failures. The identification of paths requires knowledge of the source code and data structures.
> The starting point for path testing is the flow graph. A flow graph consists of nodes representing

Question sources

Mock exam 3 · Question 39

Question 104 | Q01608

Which of the following testing is related to the boundary value analysis?

A. White & black box testing

B. White-box testing

C. Black box testing

D. None of the above

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer C: Black box testing

Boundary testing is a special case of equivalence testing, a black-box testing technique.

Why each choice is right or wrong

A - Incorrect: The cited classification does not require both white and black knowledge. See PDF p. 149, 149.

B - Incorrect: Internal path knowledge is not required by input-class edges. See PDF p. 149, 149.

C - Correct: Follows by boundary being a special case of black-box equivalence testing. See PDF p. 149, 149.

D - Incorrect: The black-box option is supported. See PDF p. 149, 149.

Primary note proof

Study-note section SE37

> Equivalence partitioning groups inputs expected to behave alike, then samples valid and invalid groups. For allowed age18..60, useful partitions are below18,18..60,above60. Boundary tests target edges and neighbors, such as 17,18,19,59,60,61, with the exact set chosen under the stated coverage scheme.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 149 | lines 18-23

> Boundary testing
> Boundary testing is a special case of equivalence testing and focuses on the conditions at the boundary of the equivalence classes. Rather than selecting any element in the equivalence class, boundary testing requires that the elements be selected from the “edges” of the equivalence class.
> The assumption behind boundary testing is that developers often overlook special cases at the boundary of the equivalence classes (e.g., 0, empty strings, year 2000).

Software Engineering module | PDF page 149 | lines 6-10

Equivalence testing is a black box testing technique that minimizes the number of test cases. The possible inputs are partitioned into equivalence classes, and a test case is selected for each class. The assumption of equivalence testing is that systems usually behave in similar ways for all members of a class. To test the behavior associated with an equivalence class, we only need to test
one member of the class. Equivalence testing consists of two steps: identification of the

Question sources

Mock exam 3 · Question 40

Question 105 | Q01636

During initial design, a component with three unrelated responsibilities is split into three cohesive components. Which concept most specifically names the resulting decomposition into responsibility-based modules, rather than a general restructuring activity?

A. Refactoring

B. Modularization

C. Information hiding

D. Refinement

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: Modularization

Separating unrelated responsibilities into cohesive components is modularization: the design is divided into modules with focused purposes. Refactoring can describe behavior-preserving restructuring of an existing solution, but that broader activity is not the specific decomposition concept requested. Information hiding concerns concealed decisions behind interfaces, while refinement is a general elaboration concept. The stated split primarily establishes better module responsibility boundaries.

Why each choice is right or wrong

A - Incorrect: Refactoring names behavior-preserving restructuring rather than the specific module decomposition principle.

B - Correct: Modularization separates responsibilities into cohesive components.

C - Incorrect: No particular hidden implementation decision defines the stated split.

D - Incorrect: Refinement is broader than the specified module-boundary decomposition.

Primary note proof

Study-note section SE22

Logical cohesion groups similar-category operations, often selected by a flag; temporal cohesion groups operations done at the same time; functional cohesion focuses on one task. Splitting an unrelated three-purpose component improves modularity. If behavior is preserved, the restructuring is refactoring; the words are not mutually exclusive answers. Smaller modules can aid understanding, independent work, reuse and maintenance, but too many interfaces also create complexity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 120 | lines 24-27

Coherence is the strength of dependencies within a subsystem. If a subsystem contains many objects that are related to each other and perform similar tasks, its coherence is high. If a subsystem
contains a number of unrelated objects, its coherence is low. A desirable property of a subsystem decomposition is that it leads to subsystems with high coherence.

Software Engineering module | PDF page 133 | lines 3-15

transformation on subsets of a particular model. Typical activities include transforming n-ary associations into binary associations, implementing binary associations as references, merging two similar classes from two different subsystems into a single class, collapsing classes with no significant behavior into attributes, splitting complex classes into simpler ones, rearranging classes and operations to increase the inheritance and packaging. During restructuring, we address design goals such as maintainability, readability, and understandability of the system model.
• Optimization. Optimization activities address performance requirements of the system model. This includes changing algorithms to respond to speed or memory requirements, reducing multiplicities in associations to speed up queries, adding redundant associations for efficiency, rearranging execution orders, adding derived attributes to improve the access time to objects and opening up the architecture, that is, adding access to lower layers because of performance requirements.

Question sources

Mock exam 2 · Question 25

Question 106 | Q01637

Which design approach characteristically starts with available reusable components and assembles them into a larger system, rather than starting from high-level decomposition?

A. Both can be used

B. Top down

C. Bottom up

D. None of the choices are correct

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Bottom up

Bottom-up design begins with existing components and assembles them into a larger solution. Top-down design instead starts with high-level structure and refines it, although it can still reuse components later. A hybrid combines the two directions, but the question asks which approach characteristically begins with the parts. Reuse by itself would not distinguish them; the explicit starting direction does.

Why each choice is right or wrong

A - Incorrect: A hybrid may use both directions but does not name the specifically described starting approach.

B - Incorrect: Top-down begins with high-level structure.

C - Correct: Bottom-up assembles existing components into the larger system.

D - Incorrect: Bottom-up directly matches, so none is false.

Primary note proof

Study-note section SE25

> Use abstraction and decomposition to manage complexity; preserve consistency, traceability and clear responsibilities. Top-down design refines high-level structure; bottom-up design assembles existing components. A hybrid combines both. SOLID includes Single Responsibility, Open-Closed, Liskov substitution, Interface Segregation and Dependency Inversion. Depend on abstractions; add compatible extensions; avoid forcing unused operations. DRY limits duplicated knowledge, KISS avoids unnecessary complexity, and YAGNI avoids speculative work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 24 | lines 20-29

> Advantages:
> 1. Reduced cost and risks
> 2. Faster delivery of the software (Accelerated development)
> Disadvantages:
> 1. Compromising requirements
> 2. Some control over the system evolution is lost as new versions of the reusable components are not under the control of the organization using them.
> 3. Increased maintenance costs
> 4. Not-invented-here syndrome
> 5. Lack of tool support

Software Engineering module | PDF page 18 | lines 13-20

System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
Software design involves identifying and describing the fundamental software system abstractions
and their relationships. The goal of the design phase is to transform the requirements specified in
the SRS document into a structure that is suitable for implementation in some programming language. In technical terms, during the design phase the software architecture is derived from the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Question sources

Mock exam 2 · Question 26

Question 107 | Q01638

Processes and threads are element types we find in _____ software structure.

A. Service software structure

B. Use software structure

C. Layer software structure

D. Concurrent software structure

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: Concurrent software structure.

SE23 explicitly puts processes and threads in concurrency views because they represent independently scheduled execution. Layer structure instead describes logical abstraction levels, while uses and service relations describe different dependencies or interfaces. Thus the concurrent structure is the offered category that directly matches these runtime execution elements.

Why each choice is right or wrong

A - Incorrect: Service structure emphasizes provided behavior or interfaces rather than processes and threads as execution elements.

B - Incorrect: Uses structure emphasizes dependencies rather than the concurrency entities named in the stem.

C - Incorrect: Layers organize logical abstraction levels rather than identifying independently scheduled execution units.

D - Correct: Concurrent structure models execution elements such as processes and threads.

Primary note proof

Study-note section SE23

A structure is the elements/relations themselves; a view represents selected aspects. Module structures show code organization, component-and-connector structures runtime interaction, and allocation structures map software to hardware, teams or environment. Processes/threads belong in concurrency views. Evaluate architectures against quality attributes and scenarios; processor allocation, decomposition and style selection can all be architectural concerns.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 121 | lines 6-15

6.1.4. Software architecture
As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept
of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
In this section, we describe a few sample architectures that can be used for different systems. This
is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
representative examples and refer the reader to the literature for more details.

Question sources

Mock exam 2 · Question 27

Question 108 | Q01639

Which statements correctly describe verification and validation? Select all that apply.

A. Verification consists of checking of documents/files and is performed by human where as validation consists of execution of program and is performed by computer.

B. All choices are correct

C. Validation assesses whether the built software meets intended stakeholder needs.

D. Verification refers to the set of activities that ensure software correctly implements the specific function.

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer C, D: Validation assesses whether the built software meets intended stakeholder needs.; Verification refers to the set of activities that ensure software correctly implements the specific function.

Verification checks whether specified functions and obligations have been implemented correctly. Validation checks whether the built product meets intended stakeholder needs. Both can use reviews, analysis and execution, so the claimed human-only versus computer-only split is false. Traceability can support these activities but does not alone establish satisfaction. If C states that validation checks whether software meets intended stakeholder needs, C and D are correct.

Why each choice is right or wrong

A - Incorrect: Both activities can use human reviews and execution; the exclusive split is false.

B - Incorrect: A is false, so all choices are not correct.

C - Correct: Validation assesses whether the product meets intended stakeholder needs.

D - Correct: Verification assesses correct implementation of specified obligations.

Primary note proof

Study-note section SE34

> Quality assurance establishes and checks processes/standards; quality control evaluates work products. An SQA plan identifies standards, responsibilities, evaluations, reviews, audits and assurance records. Quality of conformance means following the design/specification; a product can conform to a poor specification. Verification asks whether specified obligations are implemented correctly; validation asks whether the product meets intended stakeholder needs. Both can use reviews, analysis or execution, so “human-only versus computer-only” is false.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 142 | lines 11-23

> Verification attempts to find faults before any execution of the system. Verification is possible in specific cases, such as the verification of a small operating system kernel. Verification, however, has limits. It is not in a mature enough state that it can be applied to assure the quality of large complex systems. Moreover, it assumes that the requirements are correct, which is seldom the case: Assume we need to verify an operation and we know the pre- and postconditions for the operation. If we can verify that the operation provides a transformation such that, before the execution of the operation the precondition is true, and after the execution the postcondition holds,
> then the operation is correctly implemented; that is, it is verified. However, sometimes we forget important parts of the pre- or postcondition. For example, the postconditions might not state that the train should stay on the ground while moving from track 1 to track 2. Finally, verification addresses only algorithmic faults: It cannot deal with mechanical faults in the virtual machine. For
> verification, we must assume that the virtual machine on which the system is executing is correct
> and does not change during the proof.

Question sources

Mock exam 2 · Question 28

Question 109 | Q01640

Which architectural style is more suitable for if a system has to have successive transformation of streams of data?

A. Pipe and filter

B. Layered architecture

C. Client server architecture

D. All choices are correct

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer A: Pipe and filter

Pipe-and-filter architecture explicitly applies transformations to streams without user intervention.

Why each choice is right or wrong

A - Correct: Direct match to successive stream transformations. See PDF p. 126, 127, 124.

B - Incorrect: Layers organize levels, but the module does not identify them as the stream transformation style. See PDF p. 126, 127, 124.

C - Incorrect: Client/server organizes requests/services, not the requested transformation pipeline. See PDF p. 126, 127, 124.

D - Incorrect: Client/server is not equally established for the specific stream criterion. See PDF p. 126, 127, 124.

Primary note proof

Study-note section SE24

> Repository style shares central data, simplifying common access but creating bottleneck/schema dependencies. MVC separates domain model, display view and interaction controller. Client-server separates requesters from service providers; peer-to-peer permits both roles. Pipe-and-filter applies successive transformations to data streams. Three-tier separates presentation, application logic and storage; some four-tier variants split presentation client/server.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 126 | lines 7-16

Pipe and filter architecture
In the pipe and filter architecture (see Figure 6.13), subsystems process data received from a set of
inputs and send results to other subsystems via a set of outputs. The subsystems are called filters, and the associations between the subsystems are called pipes. Each filter only knows the content and the format of the data received on the input pipes, not the filters that produced them. Each filter is executed concurrently and synchronization is done via the pipes. The pipe and filter architecture is modifiable: Filters can be substituted for others or reconfigured to achieve a different purpose.
Figure 6.13 Pipe and filter architecture (UML class diagram). A Filter can have many inputs and outputs. A Pipe connects one of the outputs of a Filter to one of the inputs of another Filter.

Software Engineering module | PDF page 127 | lines 5-8

Pipe and filter architectures are suited for systems that apply transformations to streams of data without intervention by users. They are not suited for systems that require more-complex interactions between components, such as an information management system or an interactive system.

Software Engineering module | PDF page 124 | lines 8-16

Client/server architecture
In the client/server architecture (Figure 6.9), a subsystem, the server, provides servicesto instances
of other subsystems called the clients, which are responsible for interacting with theuser. The request for a service is usually done via a remote procedure call mechanism or acommon object broker (e.g., CORBA or Java RMI). Control flow in the clients and the servers isindependent except for synchronization to manage requests or to receive results.
Figure 6.9 Client/server architecture (UML class diagram).
Clients request services from one or more servers. The server has no knowledge of the client. The client/server architecture is a generalization of the repository architecture.

Question sources

Mock exam 2 · Question 29

Question 110 | Q01641

Which of the following is quality measure for software?

A. Reliability

B. Low cost

C. Productivity

D. Low development time

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer A: Reliability

Reliability is a software quality attribute characterized by behavior conformance and failure probability.

Why each choice is right or wrong

A - Correct: Explicit quality metric. See PDF p. 91, 140.

B - Incorrect: A project cost consideration rather than the offered software-behavior quality metric. See PDF p. 91, 140.

C - Incorrect: An output/effort process measure, not the specific product quality attribute. See PDF p. 91, 140.

D - Incorrect: A schedule measure rather than product reliability. See PDF p. 91, 140.

Primary note proof

Study-note section SE34

> A fault/defect is an underlying problem; an error may be an incorrect internal state; a failure is an externally observed departure from required service. Reliability is a product quality attribute; low cost or short development time are project outcomes. Tests increase confidence and reveal defects but cannot generally guarantee correctness over every possible input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 91 | lines 37-42

> • Software Metrics can be used to indicate the basic attributes of the software. The indicative metrics, not exhaustive, are
> • Size - Line of code ,Function point count
> • Quality- Reliability, Accuracy, Dependability, and these are measured through some measures like errors, failures and number of runs between failures ,etc.
> • Execution Time -process time of execution of a critical process and many more.

Software Engineering module | PDF page 140 | lines 11-18

> Reliability is a measure of success with which the observed behavior of a system conforms to the specification of its behavior. Software reliability is the probability that a software system will not cause the failure of the system for a specified time under specified conditions. Failure is any deviation of the observed behavior from the specified behavior. An error means the system is in a state such that further processing by the system will lead to a failure, which causes the system to
> deviate from its intended behavior. A fault, also called defect or bug, is the mechanical or algorithmic cause of an error. The goal of testing is to maximize the number of discovered faults, which then allows developers to correct them and increase the reliability of the system.

Question sources

Mock exam 2 · Question 30

Question 111 | Q01642

A project manager is using PERT to estimate the duration of a project activity. The optimistic time (a) is 3 days, the most likely time (m) is 6 days, and the pessimistic time (b) is 15 days. What is the expected time (TE) for the activity?

A. 8 days

B. 10 days

C. 7 days

D. 9 days

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer C: 7 days.

Use TE=(a+4m+b)/6 from SE30 with a=3, m=6 and b=15. The numerator is 3+24+15=42, so TE=42/6=7 days. A simple average would give 8, but that discards the fourfold weighting of the most-likely estimate. The expected duration is an estimate, not a guaranteed actual completion time.

Why each choice is right or wrong

A - Incorrect: Eight is the simple mean of 3, 6 and 15, not the weighted PERT estimate.

B - Incorrect: Ten does not result from substituting the three given values into the PERT formula.

C - Correct: The weighted sum is 42 and dividing by six gives 7 days.

D - Incorrect: Nine does not equal the PERT weighted expected time for these estimates.

Primary note proof

Study-note section SE30

> Traditional CPM uses point durations; PERT models uncertainty with optimistic a, most-likely m and pessimistic b. The classic estimate is TE=(a+4m+b)/6 and variance=((b-a)/6)^2. For 3,6,15 days, TE=42/6=7 days and variance=4 days². Do not replace the weighted estimate by a simple average. These formulas are approximations, not guaranteed actual completion times.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 3-12

> 4.6. Scheduling, Organization and Team Structures
> Project-task scheduling is an important project planning activity. It involves deciding which tasks would
> be taken up when. In order to schedule the project activities, a software project manager needs to do
> the following:

• Identify all the tasks needed to complete the project and Break down large tasks into small activities.
• Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
• Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Mock exam 2 · Question 31

Question 112 | Q01643

Which of the following is NOT a characteristic of a critical path in a CPM network diagram for software development?

A. Any delay in tasks on this path delays the entire project.

B. It identifies tasks with strict dependencies.

C. It represents the longest sequence of tasks.

D. It has the most slack (buffer time) compared to other paths.

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer D: It has the most slack (buffer time) compared to other paths.

The critical path is the chain determining project duration. The maximum-slack claim contradicts this: freely postponable tasks would not determine the earliest completion under a fixed network.

Why each choice is right or wrong

A - Incorrect: Delaying a controlling chain delays completion. See PDF p. 95.

B - Incorrect: Dependency order constrains the chain, though noncritical tasks can also have strict dependencies. See PDF p. 95.

C - Incorrect: The duration-determining chain is the longest required chain in the fixed-duration network. See PDF p. 95.

D - Correct: Maximum slack is inconsistent with a path that controls completion; no resource constraints are assumed. See PDF p. 95.

Primary note proof

Study-note section SE29

In a dependency network, ES=max predecessor EF and EF=ES+duration; start tasks have ES=0 unless another constraint applies. Project earliest finish is the maximum terminal EF. Backward, LF=min successor LS and LS=LF-duration, starting from the required finish. Total

float/slack=LS-ES=LF-EF. A critical path has the longest total duration and normally zero float in an unconstrained schedule.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 3-12

4.6. Scheduling, Organization and Team Structures
Project-task scheduling is an important project planning activity. It involves deciding which tasks would
be taken up when. In order to schedule the project activities, a software project manager needs to do
the following:
• Identify all the tasks needed to complete the project and Break down large tasks into small activities.
• Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
• Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Mock exam 2 · Question 32

Question 113 | Q01644

A zero-duration start A leads to parallel paths B(5)→C(6) and D(8)→E(3), both ending at a zero-duration finish. Which critical-path statements are true? Select all that apply. Multiple critical paths are allowed.

A. A→B→C is a critical path with total duration 11 days.

B. A→D→E is a critical path with total duration 11 days.

C. A→D→E is a critical path with total duration 14 days.

D. A→B→C is a critical path with total duration 14 days.

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer A, B: A→B→C is a critical path with total duration 11 days.; A→D→E is a critical path with total duration 11 days.

The B-C branch totals 5+6=11 days and the D-E branch totals 8+3=11 days. With zero-duration start and finish, both are equally longest complete paths, so both are critical and the project duration is eleven days. There is no rule requiring a unique critical path. The alternatives using fourteen days miscalculate the corresponding branch totals.

Why each choice is right or wrong

A - Correct: The B-C branch totals eleven and is critical.

B - Correct: The D-E branch also totals eleven and is critical.

C - Incorrect: Eight plus three is eleven, not fourteen.

D - Incorrect: Five plus six is eleven, not fourteen.

Primary note proof

Study-note section SE29

> In a dependency network, ES=max predecessor EF and EF=ES+duration; start tasks have ES=0 unless another constraint applies. Project earliest finish is the maximum terminal EF. Backward, LF=min successor LS and LS=LF-duration, starting from the required finish. Total float/slack=LS-ES=LF-EF. A critical path has the longest total duration and normally zero float in an unconstrained schedule.

Study-note section SE29

> If A takes 5 days and B=3,C=4 both follow A, project duration is 5+max(3,4)=9 days. B finishes at 8, but the project waits for C at 9. If paths B5+C6 and D8+E3 share a zero-duration start/end, both total 11; there are two critical paths, not one.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 3-12

> 4.6. Scheduling, Organization and Team Structures
> Project-task scheduling is an important project planning activity. It involves deciding which tasks would
> be taken up when. In order to schedule the project activities, a software project manager needs to do
> the following:
> • Identify all the tasks needed to complete the project and Break down large tasks into small activities.
> • Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
> • Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Mock exam 2 · Question 33

Question 114 | Q01645

How can market dynamics influence the evolution of requirements?

A. By introducing new competitors with innovative products

B. By stabilizing existing market conditions

C. By reducing the need for frequent updates

D. By minimizing consumer demands

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: By introducing new competitors with innovative products

Product competition is a business risk. A competitor introducing a product can alter customer needs and thus motivate evolution.

Why each choice is right or wrong

A - Correct: Direct concrete competitive stimulus consistent with the risk table on PDF p.97. See PDF p. 96, 41.

B - Incorrect: Stabilization does not supply the change stimulus described. See PDF p. 96, 41.

C - Incorrect: Reduced updates does not explain new competitive requirements. See PDF p. 96, 41.

D - Incorrect: Minimizing demands is not the described competitive pressure. See PDF p. 96, 41.

Primary note proof

Study-note section SE11

> Correct requirements reflect actual stakeholder needs. Complete requirements address the agreed scope, normal and exceptional behavior. Consistency excludes contradictions. Clarity/unambiguity gives one intended interpretation; feasibility/realism means achievable within constraints. Verifiability means objective evidence can check fulfillment. Replace “fast for most users” with a defined metric, workload and threshold.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 96 | lines 12-21

> 4.7. Risk Management
> • Risk management is concerned with identifying risks and drawing up plans to minimise their effect on a project.
> • A risk is a probability that some adverse circumstance will occur
> - Project risks affect schedule or resources;
> - Product risks affect the quality or performance of the software being developed;
> - Business risks affect the organisation developing or procuring the software.
> Risk Management Process

- Risk identification- Identify project, product and business risks;
- Risk analysis -Assess the likelihood and consequences of these risks;

Software Engineering module | PDF page 41 | lines 16-17

✓ Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous and the software must evolve to remain useful.

Software Engineering module | PDF page 97 | lines 30-33

Technology change Business The underlying technology on which the system is built is superseded by new technology.
Product competition Business A competitive product is marketed before the system is completed.

Question sources

Mock exam 2 · Question 34

Question 115 | Q01646

The static nature of the system is represented by which model?

A. Context Model

B. Data Model

C. Structural Model

D. Behavioral Model

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Structural Model

The class/object model describes structure rather than temporal behavior; structural model is the offered general category.

Why each choice is right or wrong

A - Incorrect: Context concerns system boundaries, not all internal static structure. See PDF p. 51, 101.

B - Incorrect: Data is one aspect of structure but narrower than the overall static model. See PDF p. 51, 101.

C - Correct: Best matches the system's static classes/relationships. See PDF p. 51, 101.

D - Incorrect: Behavioral models describe interactions and changes over time. See PDF p. 51, 101.

Primary note proof

Study-note section SE17

A class box lists name, attributes and operations; + public, - private, # protected and ~ package indicate visibility. A class diagram shows static types and relationships, while an object diagram shows particular instances/links. UML is maintained and standardized by OMG; it is a modeling language, not a lifecycle process. Structural things describe conceptual or physical structural elements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 51 | lines 14-24

2.3.2. Class Diagrams
Classes and objects
Class diagrams describe the structure of the system in terms of classes and objects. Classes are abstractions that specify the attributes and behavior of a set of objects. Objects are entities that encapsulate state and behavior. Each object has an identity: It can be referred individually and is distinguishable from other objects.
In UML, classes and objects are depicted by boxes including three compartments. The top compartment displays the name of the class or object. The center compartment displays its attributes, and the bottom compartment displays its operations. The attribute and operation compartment can be omitted for clarity. Object names are underlined to indicate that they are instances. By convention, class names start with an uppercase letter. Objects in object diagrams

Software Engineering module | PDF page 101 | lines 5-6

In UML, the functional model is represented with use case diagrams, the object model with class diagrams, and the dynamic model with statechart and sequence diagrams.

Question sources

Mock exam 2 · Question 35

Question 116 | Q01647

How does backward traceability help in requirements management?

A. By identifying redundant requirements

B. By mapping requirements to design documents

C. By ensuring that each requirement can be traced back to its source

D. By linking requirements to future project phases

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Trace each requirement back to its source and rationale.

SE11 directs requirements backward to their source or rationale and forward to design, code and tests. Backward traceability therefore establishes why a requirement exists and where it came from. It can assist reviews, but redundancy detection is not its definition. Design mappings and future phases primarily describe the forward direction.

Why each choice is right or wrong

A - Incorrect: Redundancy detection can benefit from traceability, but it is not the definition of backward tracing.

B - Incorrect: Mapping toward design documents is forward traceability from the requirement.

C - Correct: Backward tracing establishes the source or rationale from which each requirement originated.

D - Incorrect: Links toward later project artifacts describe forward rather than backward traceability.

Primary note proof

Study-note section SE11

> Trace each requirement backward to its source/rationale and forward to design, code and tests. Record dependencies among requirements. A traceability matrix links these artifacts and helps impact analysis; it is not just a cost/schedule ledger. Stakeholders must negotiate conflicting requirements rather than independently approving incompatible demands. Scope, understanding and volatility are distinct elicitation difficulties.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 70 | lines 3-16

> 3.2.5. Verifiability and Traceability
> Two more desirable properties of a system specification are that it be verifiable and traceable. The
> specification is verifiable if, once the system is built; a repeatable test can be designed to demonstrate that the system fulfills the requirement. For example, a mean time to failure of a hundred years for SatWatch would be difficult to achieve (assuming it is realistic in the first place).
> The following requirements are additional examples of non-verifiable requirements:
> • The product shall have a good user interface (good is not defined).
> • The product shall be error free (requires large amount of resources to establish).
> • The product shall respond to the user with 1 second for most cases (“most cases” is not defined).
> A system specification is traceable if each system function can be traced to its corresponding set of requirements. Traceability is not a constraint on the content of the specification, but rather, on its organization. Traceability facilitates the development of tests and the systematic validation of the design against the requirements.

Question sources

Mock exam 2 · Question 36

Question 117 | Q01648

Which artifact primarily serves project budgeting rather than specifying, planning or tracing tests?

A. Test strategy

B. Test Case

C. Project-wide cost estimate

D. Requirements Traceability Matrix [RTM]

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer C: Project-wide cost estimate

A test strategy describes how testing will be approached, a test case supplies actions or inputs with expected results, and an RTM links requirements to tests to support coverage and impact analysis. A project-wide cost estimate instead budgets the broader project. It may include testing costs and inform a test plan, but its primary purpose is project planning rather than specifying or tracing tests. The distractor uses that explicit distinction.

Why each choice is right or wrong

A - Incorrect: A test strategy defines the intended testing approach.

B - Incorrect: A test case specifies inputs or actions and expected results.

C - Correct: A project-wide cost estimate is a budgeting artifact, not primarily a testing specification or trace.

D - Incorrect: An RTM links requirements to tests and supports test coverage.

Primary note proof

Study-note section SE39

> Plan objectives, scope, resources, schedule, risks and entry/exit criteria. Analyze the test basis for testable features and defects; design cases/expected results; implement data, scripts and environment; execute and compare; report defects; evaluate completion.

Study-note section SE11

> Trace each requirement backward to its source/rationale and forward to design, code and tests. Record dependencies among requirements. A traceability matrix links these artifacts and helps impact analysis; it is not just a cost/schedule ledger.

Study-note section SE28

Planning is continuous: identify scope/deliverables, decompose work, estimate effort/duration/cost, assign resources, sequence dependencies and revise as evidence changes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Software Engineering module | PDF page 90 | lines 7-24

4.2. Project Planning
• Probably the most time-consuming project management activity.
• Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
• Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
The project plan sets out:
• The resources available to the project;
• The work breakdown;
• A schedule for the work.
Project Plan Structure
• Introduction.
• Project organisation.
• Risk analysis.
• Hardware and software resource requirements.
• Work breakdown.
• Project schedule.
• Monitoring and reporting mechanisms.

Question sources

Mock exam 2 · Question 37

Question 118 | Q01649

Degree to which design specifications are followed in manufacturing the product is called

A. Quality Control

B. None of the mentioned

C. Quality of conformance

D. Quality Assurance

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer C: Quality of conformance.

SE34 defines quality of conformance as following the design or specification. That is exactly the degree asked for in the stem. Quality control evaluates work products and quality assurance establishes and checks processes, but neither is the name of this conformance property. A conforming product may still follow an inadequate specification.

Why each choice is right or wrong

A - Incorrect: Quality control evaluates products rather than naming their degree of adherence to the design.

B - Incorrect: The exact property is offered as quality of conformance.

C - Correct: Conformance measures how faithfully the product follows its design or specification.

D - Incorrect: Quality assurance concerns processes and standards rather than the named degree of product conformance.

Primary note proof

Study-note section SE34

> Quality assurance establishes and checks processes/standards; quality control evaluates work products. An SQA plan identifies standards, responsibilities, evaluations, reviews, audits and assurance records. Quality of conformance means following the design/specification; a product can conform to a poor specification. Verification asks whether specified obligations are implemented correctly; validation asks whether the product meets intended stakeholder needs. Both can use reviews, analysis or execution, so “human-only versus computer-only” is false.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 99 | lines 14-19

> 4.8. Quality Assurance Monitoring Plans
> Quality assurance consists of the auditing and reporting functions of management. The goal of quality assurance is to provide management with the data necessary to be informed about product

quality, thereby gaining insight and confidence that product quality is meeting its goals. Of course,
if the data provided through quality assurance identify problems, it is management's responsibility
to address the problems and apply the necessary resources to resolve quality issues.

Software Engineering module | PDF page 140 | lines 11-18

Reliability is a measure of success with which the observed behavior of a system conforms to the specification of its behavior. Software reliability is the probability that a software system will not cause the failure of the system for a specified time under specified conditions. Failure is any deviation of the observed behavior from the specified behavior. An error means the system is in a state such that further processing by the system will lead to a failure, which causes the system to
deviate from its intended behavior. A fault, also called defect or bug, is the mechanical or algorithmic cause of an error. The goal of testing is to maximize the number of discovered faults, which then allows developers to correct them and increase the reliability of the system.

Question sources

Mock exam 2 · Question 38

Question 119 | Q01650

Which of the below testing is related to Non-functional testing?

A. Performance Testing

B. Unit Testing

C. Black-box Testing

D. None of the choices are correct

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer A: Performance Testing

Performance testing explicitly checks nonfunctional requirements and additional design goals.

Why each choice is right or wrong

A - Correct: Directly tests nonfunctional qualities. See PDF p. 144.

B - Incorrect: A test level that can use several techniques and requirement types. See PDF p. 144.

C - Incorrect: A knowledge-based technique category, not specifically nonfunctional scope. See PDF p. 144.

D - Incorrect: The performance option is supported. See PDF p. 144.

Primary note proof

Study-note section SE36

Black-box techniques derive cases from specified behavior; white-box techniques derive them from internal structure; grey-box uses partial internal knowledge. Regression testing checks for unintended effects after change; confirmation testing checks the particular defect repair. Exploratory testing combines learning, design and execution and is experience based. Manual/automated describe execution support, not a compulsory black-/white-box classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Question sources

Mock exam 2 · Question 39

Question 120 | Q00034B

First design the very high level structure of the system. Then gradually work down to detailed decisions about low-level constructs. Which design approach follows the given strategy?

A. Bottom-up

B. System design

C. Top-down

D. Hybrid

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Top-down design.

The described direction starts with high-level structure and refines it into lower-level decisions. SE25 explicitly calls that top-down design. Bottom-up starts from existing components and assembles them; a hybrid combines directions. System design names a design activity or level, not the specific refinement direction requested by the question.

Why each choice is right or wrong

A - Incorrect: Bottom-up proceeds from components toward a larger structure, reversing the described direction.

B - Incorrect: System design names an architectural activity rather than the specific high-to-low refinement approach.

C - Correct: Top-down begins with high-level structure and progressively refines implementation details.

D - Incorrect: A hybrid combines both directions; the stem describes only top-down refinement.

Primary note proof

Study-note section SE25

> Use abstraction and decomposition to manage complexity; preserve consistency, traceability and clear responsibilities. Top-down design refines high-level structure; bottom-up design assembles existing components. A hybrid combines both. SOLID includes Single Responsibility, Open-Closed, Liskov substitution, Interface Segregation and Dependency Inversion. Depend on abstractions; add compatible extensions; avoid forcing unused operations. DRY limits duplicated knowledge, KISS avoids unnecessary complexity, and YAGNI avoids speculative work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 118 | lines 11-16

> 6.1.1. Subsystems and classes
> In order to reduce the complexity of the application domain, we identified smaller parts called classes and organized them into packages. Similarly, to reduce the complexity of the solution domain, we decompose a system into simpler parts, called subsystems, which are made of a number of solution domain classes. In the case of complex subsystems, we recursively apply this principle and decompose a subsystem into simpler subsystems (see Figure 6.2).

Software Engineering module | PDF page 18 | lines 13-20

> System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
> Software design involves identifying and describing the fundamental software system abstractions
> and their relationships. The goal of the design phase is to transform the requirements specified in
> the SRS document into a structure that is suitable for implementation in some programming language. In technical terms, during the design phase the software architecture is derived from

the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 34

Question 121 | SM0006

If you plan a software test to be done by clients to demonstrate that the system satisfies their requirements, which phase of software testing is appropriate?

A. Acceptance testing

B. Integration testing

C. System testing

D. Unit testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer A: Acceptance testing

Acceptance testing checks requirements against the project agreement and is performed by the client, matching both actor and purpose in the stem.

Why each choice is right or wrong

A - Correct: Direct client agreement/acceptance check. See PDF p. 145, 144.

B - Incorrect: Combines individually tested components. See PDF p. 145, 144.

C - Incorrect: Tests the complete system, but the specified client acceptance purpose identifies the narrower level. See PDF p. 145, 144.

D - Incorrect: Isolates building blocks, not client acceptance. See PDF p. 145, 144.

Primary note proof

Study-note section SE35

Acceptance testing supports the customer/user's acceptance decision against agreed needs and criteria. Alpha typically happens in a controlled developer setting with selected users; beta uses external users in their environments. None of these levels is universally the longest. Unit is a level, while white-box is a technique; a unit can be tested using black-box cases too.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 145 | lines 3-5

o Acceptance testing and installation testing check the requirements against the project agreement and should be done by the client, if necessary with support by the developers.

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Question sources

MOE Computer Science Exit Exam · Question 2-2

National Computer Science Exit Exam · Question 46

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 1

Question 122 | SM0027

Assume you make modification to your software. Then you want to make sure that the modification
has not introduced new errors. Which software testing level is required?

A. Regression testing

B. Static testing

C. Beta testing

D. Dynamic testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer A: Regression testing.

Regression testing means checking for unintended effects after a software change. That matches the stated goal of finding newly introduced errors. Static and dynamic describe execution-related approaches rather than this change-focused purpose; beta describes an external-user test setting. Although the stem says level, regression is more accurately a testing purpose/type.

Why each choice is right or wrong

A - Correct: Regression testing checks whether a modification damages behavior that previously worked.

B - Incorrect: Static testing examines artifacts without execution and does not specifically define the change-impact purpose.

C - Incorrect: Beta testing involves external users in their environments rather than specifically checking every change's unintended effects.

D - Incorrect: Dynamic testing means executing software; it is broader than the regression purpose described.

Primary note proof

Study-note section SE36

> Black-box techniques derive cases from specified behavior; white-box techniques derive them from internal structure; grey-box uses partial internal knowledge. Regression testing checks for unintended effects after change; confirmation testing checks the particular defect repair. Exploratory testing combines learning, design and execution and is experience based. Manual/automated describe execution support, not a compulsory black-/white-box classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 147 | lines 14-18

> • Unit testing, which finds faults by isolating an individual component using test stubs and drivers and by exercising the component using a test case
> • Integration testing, which finds faults by integrating several components together
> • System testing, which focuses on the complete system, its functional and nonfunctional requirements, and its target environment

Software Engineering module | PDF page 142 | lines 5-10

> Configuration management avoids faults caused by undisciplined change in the system models. For example, it is a common mistake to a change a subsystem interface without notifying all developers of calling components. Configuration management can make sure that, if analysis models and code are becoming inconsistent with one another, analysts and implementor are notified. The assumption behind configuration management is that the system contains many less
> faults if change is controlled.

Software Engineering module | PDF page 147 | lines 3-8

• A test stub is a partial implementation of components on which the tested component depends. A test driver is a partial implementation of a component that depends on the tested component. Test stubs and drivers enable components to be isolated from the rest of the system for testing.
• A correction is a change to a component. The purpose of a correction is to repair a fault. Note that a correction can introduce new faults.

Question sources

MOE Computer Science Exit Exam · Question 8-2

National Computer Science Exit Exam · Question 54

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 2

Question 123 | SM0066

After deep analysis if your software requires major change to the architecture of the system, which
strategy of software change should be applied?

A. Preventative maintenance

B. Architectural transformation

C. Software reengineering

D. Software maintenance

Hint

Classify the motive for the change before its implementation: repair, environment, useful capability or reduced future risk.

Answer B: Architectural transformation.

SE40 names major restructuring reengineering and identifies architectural transformation as the more specific term when the architecture itself changes. The stem expressly requires major architecture change, making B the best match. Maintenance and reengineering are broader categories, while preventive maintenance describes a motive and need not involve architectural restructuring.

Why each choice is right or wrong

A - Incorrect: Preventive maintenance reduces future risk but does not specifically denote a major architectural change.

B - Correct: Architectural transformation directly names the major architecture change required by the scenario.

C - Incorrect: Reengineering is a broader restructuring category; architectural transformation is more specific here.

D - Incorrect: Maintenance is the broad activity family rather than the particular architecture-change strategy.

Primary note proof

Study-note section SE40

A typical evolution sequence is change request→impact analysis→release planning→implementation/testing→release. Among a list lacking the request step, impact analysis may be the earliest listed technical activity. Major restructuring is reengineering; if the architecture itself changes, architectural transformation is the more specific description. Markets, technology, stakeholders and funding can all change scope or priorities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 121 | lines 6-15

6.1.4. Software architecture
As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept
of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
In this section, we describe a few sample architectures that can be used for different systems. This
is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
representative examples and refer the reader to the literature for more details.

Software Engineering module | PDF page 19 | lines 24-29

5. Operation and maintenance: Normally (although not necessarily), this is the longest life cycle phase. Maintenance of a typical software product requires much more than the effort necessary to
develop the product itself. The system is installed and put into practical use. Maintenance involves
correcting errors which were not discovered in earlier stages of the life cycle, improving the implementation of system units and enhancing the system ‘s services as new requirements are discovered.

Question sources

MOE Computer Science Exit Exam · Question 19-2

Computer Science Exit Exam practice collection · Question 3

Question 124 | SM0078

Which one of the following does NOT show a non-functional requirement of software systems?

A. Response time

B. Memory requirement

C. Reliability

D. Displaying information

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer D: Displaying information

Response time, memory constraints and reliability are nonfunctional qualities. Displaying information specifies a service rather than a quality bound.

Why each choice is right or wrong

A - Incorrect: Quantitative speed constraint. See PDF p. 65, 66, 127.

B - Incorrect: Space/resource constraint. See PDF p. 65, 66, 127.

C - Incorrect: Dependability quality. See PDF p. 65, 66, 127.

D - Correct: Observable function of presenting information. See PDF p. 65, 66, 127.

Primary note proof

Study-note section SE09

> A functional requirement states an observable service or behavior, including reactions to inputs, errors and forbidden actions. “When the sensor detects intrusion, send an alert” is functional. “Only a manager may approve refunds” also specifies permitted/prohibited behavior. A negative statement is not automatically nonfunctional.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 65 | lines 19-23

> 3.2.1. Functional Requirements
> Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. For example, the following is an example of functional requirements for SatWatch, a watch that resets itself without user intervention:

Software Engineering module | PDF page 66 | lines 14-18

> 3.2.2. Non-Functional and Pseudo-Requirements
> Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Nonfunctional requirements include

quantitative constraints, such as response time (i.e., how fast the system reacts to user commands)
or accuracy (i.e., how precise are the system's numerical answers). The following are the

Software Engineering module | PDF page 127 | lines 10-28

6.2.1. Identifying design goals
The definition of design goals is the first step of system design. It identifies the qualities that our system should focus on. Many design goals can be inferred from the nonfunctional requirements or from the application domain. Others will have to be elicited from the client. It is, however, necessary to state them explicitly such that every important design decision can be made consistently following the same set of criteria.
For example, in the light of the nonfunctional requirements for MyTrip, we identify reliability and
fault tolerance to connectivity loss as design goals. We then identify security as a design goal, as numerous drivers will have access to the same trip planning server. We add modifiability as a design goal, as we want to provide the ability for drivers to select a trip planning service of their choice. The following box summarizes the design goals we identified.
In general, we can select design goals from a long list of highly desirable qualities. Tables 5-1-5-5 list a number of possible design criteria. These criteria are organized into five groups: performance, dependability, cost, maintenance, and end user criteria. Performance, dependability,
and end user criteria are usually specified in the requirements or inferred from the application domain. Cost and maintenance criteria are dictated by the customer and the supplier.
Performance criteria (Table 5-1) include the speed and space requirements imposed on the system. Should the system be responsive or should it accomplish a maximum number of tasks? Is memory space available for speed optimizations or should memory be used sparingly?

Question sources

MOE Computer Science Exit Exam · Question 22-3

National Computer Science Exit Exam · Question 45

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 4

Question 125 | SM0079

Which one of the following is false about project planning stage?

A. Risk analysis is done during project planning

B. Preparing time schedule is done during project planning

C. Cost estimation is done during project planning

D. Project planning is a one-time task in software development life cycle

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer D: Project planning is a one-time task in software development life cycle

Planning is explicitly continuous from initial concept through delivery and revised as information changes. Therefore the one-time statement is false.

Why each choice is right or wrong

A - Incorrect: Risk analysis is in the plan structure. See PDF p. 90.

B - Incorrect: Schedule is in the project plan. See PDF p. 90.

C - Incorrect: Plans concern schedule and budget, so cost estimation belongs in planning. See PDF p. 90.

D - Correct: Direct contradiction of continuous revision. See PDF p. 90.

Primary note proof

Study-note section SE28

> Planning is continuous: identify scope/deliverables, decompose work, estimate effort/duration/cost, assign resources, sequence dependencies and revise as evidence changes. A WBS is hierarchical and covers project scope without adding unrelated work. Tools help coordinate ownership, dependencies and progress; they do not replace judgment or automatically fix bugs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 90 | lines 7-24

> 4.2. Project Planning
> • Probably the most time-consuming project management activity.
> • Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
> • Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
> The project plan sets out:
> • The resources available to the project;
> • The work breakdown;
> • A schedule for the work.
> Project Plan Structure
> • Introduction.
> • Project organisation.
> • Risk analysis.
> • Hardware and software resource requirements.
> • Work breakdown.

• Project schedule.
• Monitoring and reporting mechanisms.

Question sources

MOE Computer Science Exit Exam · Question 22-4

National Computer Science Exit Exam · Question 14

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 5

Question 126 | SM0090

Which phase of software development process refers to modification of software to reflect changing customer and market requirements?

A. Validation

B. Evolution

C. Development

D. Specification

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Evolution

Changing existing software to meet new requirements is software evolution.

Why each choice is right or wrong

A - Incorrect: Checks conformance and customer expectations. See PDF p. 41.

B - Correct: Exact ongoing change activity. See PDF p. 41.

C - Incorrect: Produces a software implementation. See PDF p. 41.

D - Incorrect: Defines required services and constraints. See PDF p. 41.

Primary note proof

Study-note section SE01

Software engineering applies systematic methods to development, operation and maintenance. A software product includes executable code and relevant requirements, designs, test assets, configuration and user documentation. A process defines activities, ordering, roles, products and entry/exit conditions. Its four broad activities are specification, design/implementation, validation and evolution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 41 | lines 16-17

> ✓ Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous and the software must evolve to remain useful.

Question sources

MOE Computer Science Exit Exam · Question 25-3

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 6

Computer Science Exit Exam practice collection · Question 11

Question 127 | SM0203

When is it appropriate to use the Spiral model?

A. Where business modeling is clearly understood and in use

B. When it is too risky to develop the overall system as one block

C. When requirements are understood early but can change during the project's life cycle

D. When requirements are unstable and constantly changing

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: When major risks make one-block development unsafe.

SE06 uses explicit risk assessment and reduction to distinguish spiral development from other iterative approaches. B directly states that risk makes a single overall commitment inappropriate. Changing or unstable requirements can occur in several processes and do not alone identify spiral; clear business modeling likewise does not supply the distinctive risk-based reason.

Why each choice is right or wrong

A - Incorrect: Understanding business modeling alone does not identify the need for risk-driven spiral cycles.

B - Correct: Important risks justify staged investigation and commitment rather than building the entire system in one block.

C - Incorrect: Possible requirement changes fit multiple processes and omit spiral's distinguishing explicit risk emphasis.

D - Incorrect: Unstable requirements alone do not uniquely select spiral over other adaptive approaches.

Primary note proof

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Study-note section SE06

> Use risk emphasis to distinguish spiral from a generic changing-requirements scenario.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 96 | lines 12-21

> 4.7. Risk Management
> • Risk management is concerned with identifying risks and drawing up plans to minimise their effect on a project.
> • A risk is a probability that some adverse circumstance will occur
> - Project risks affect schedule or resources;
> - Product risks affect the quality or performance of the software being developed;
> - Business risks affect the organisation developing or procuring the software.
> Risk Management Process
> - Risk identification- Identify project, product and business risks;
> - Risk analysis -Assess the likelihood and consequences of these risks;

Software Engineering module | PDF page 20 | lines 18-22

> the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
> In principle, the waterfall model should only be used when the requirements are well understood
> and unlikely to change radically during system development. However, the waterfall model

Software Engineering module | PDF page 22 | lines 8-12

> This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
> also provide a platform for evaluation by the user.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 1-3

Computer Science Exit Exam practice collection · Question 14

Question 128 | SM0235

A module is said to have logical cohesion if:

A. All the functions of the module are executed within the same time span

B. All elements of the module depend on their interface complexity

C. It performs a set of tasks that relate to each other very loosely

D. All elements of the module perform similar operations

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: The module groups similar-category operations.

SE22 defines logical cohesion as grouping operations in a similar category, often selected through a flag. D matches that shared logical category. A instead describes temporal cohesion, where operations occur at the same time. Interface complexity concerns dependencies, and a loose collection of unrelated tasks does not state the defining category-based relationship.

Why each choice is right or wrong

A - Incorrect: Grouping operations because they occur at the same time is temporal cohesion.

B - Incorrect: Interface complexity concerns dependencies and does not define logical cohesion inside a module.

C - Incorrect: Loose task relatedness alone does not establish the shared operation category defining logical cohesion.

D - Correct: Grouping similar-category operations is the defining feature of logical cohesion.

Primary note proof

Study-note section SE22

> Logical cohesion groups similar-category operations, often selected by a flag; temporal cohesion groups operations done at the same time; functional cohesion focuses on one task. Splitting an unrelated three-purpose component improves modularity. If behavior is preserved, the restructuring is refactoring; the words are not mutually exclusive answers. Smaller modules can aid understanding, independent work, reuse and maintenance, but too many interfaces also create complexity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 120 | lines 24-27

> Coherence is the strength of dependencies within a subsystem. If a subsystem contains many objects that are related to each other and perform similar tasks, its coherence is high. If a subsystem

contains a number of unrelated objects, its coherence is low. A desirable property of a subsystem decomposition is that it leads to subsystems with high coherence.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 11-3

Computer Science Exit Exam practice collection · Question 17

Question 129 | SM0244

Which software engineering principles are used to help reduce programming complexity?

A. Structured and unstructured

B. Object Oriented Design

C. Abstraction and Decomposition

D. Functional and Non-functional

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Abstraction and Decomposition

Abstraction lets developers reason about objects at a higher level while decomposition divides a complex system into simpler subsystems; both directly control complexity.

Why each choice is right or wrong

A - Incorrect: Structured/unstructured names do not identify the paired techniques. See PDF p. 31, 118.

B - Incorrect: OO design employs these mechanisms but is the broader methodology. See PDF p. 31, 118.

C - Correct: Combines the two explicit complexity-management mechanisms. See PDF p. 31, 118.

D - Incorrect: These classify requirements, not techniques for decomposing a program. See PDF p. 31, 118.

Primary note proof

Study-note section SE22

Cohesion measures how well elements inside a module serve a related purpose; coupling measures dependency between modules. Seek high cohesion and low coupling. Shared mutable data and knowledge of another module's internal representation increase coupling. Information hiding exposes stable operations while concealing changeable decisions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 31 | lines 19-31

1.3.1. Why an object oriented
• Object oriented systems are
o Easier to adapt to changes
o More robust
o Easier to maintain
o Promote greater design and code reuse
o Creates modules of functionality
• REASONS FOR WORKING OF OBJECT ORIENTED SYSTEMS:
Higher level of abstraction
The object-oriented approach supports abstraction at the object level. Since objects encapsulate both data (attributes) and functions (methods), they work at a higher level of abstraction. The development can proceed at the object level and ignore the rest of the system for as long as necessary. This makes designing, coding, testing, and maintaining the system much simpler.

Software Engineering module | PDF page 118 | lines 11-16

6.1.1. Subsystems and classes
In order to reduce the complexity of the application domain, we identified smaller parts called classes and organized them into packages. Similarly, to reduce the complexity of the solution domain, we decompose a system into simpler parts, called subsystems, which are made of a number of solution domain classes. In the case of complex subsystems, we recursively apply this principle and decompose a subsystem into simpler subsystems (see Figure 6.2).

Question sources

Computer Science Exit Exam · Sene 2016 · Question 14-3

Computer Science Exit Exam practice collection · Question 15

Question 130 | SM0271

Which listed model organizes repeated cycles around risk assessment and can select prototyping, incremental or sequential activities as appropriate within those cycles?

A. Spiral model

B. Prototyping model

C. Classical Waterfall model

D. Iterative Waterfall model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A: Spiral model

Spiral development repeatedly identifies objectives, assesses and reduces risks, develops an appropriate result and plans further commitment. Its cycles can incorporate prototyping, incremental work or sequential activities according to the risks. The other listed models name particular exploratory or staged organizations rather than this risk-driven choice framework. This explains the conventional general-model reading without claiming that every process is formally a specialization of spiral.

Why each choice is right or wrong

A - Correct: Spiral selects cycle work through risk assessment.

B - Incorrect: Prototyping is one learning activity that a spiral cycle may use.

C - Incorrect: Classical waterfall is a planned sequential organization.

D - Incorrect: Iterative waterfall adds feedback without defining the risk-driven cycle.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 20 | lines 18-22

> the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
> In principle, the waterfall model should only be used when the requirements are well understood
> and unlikely to change radically during system development. However, the waterfall model

Software Engineering module | PDF page 22 | lines 8-12

> This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user

and
also provide a platform for evaluation by the user.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 23-2

Computer Science Exit Exam practice collection · Question 16

Question 131 | SM0293

Which system requirement specification includes human-computer interface issues from the system?

A. Functional requirements of the system

B. Non-functional requirements of the system

C. Goals of implementation

D. System Testing

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: Non-functional requirements of the system

User-interface look/feel and user expertise are explicitly nonfunctional concerns, under user interface and human factors. Particular interaction services can still be functional; the question is read as interface quality issues.

Why each choice is right or wrong

A - Incorrect: Specific interface functions can be functional, but this is not the broad human-factor quality category. See PDF p. 78, 66.

B - Correct: Explicitly includes user interface and human factors. See PDF p. 78, 66.

C - Incorrect: Implementation goals are not the stated requirement category. See PDF p. 78, 66.

D - Incorrect: Testing is an activity, not a requirement category. See PDF p. 78, 66.

Primary note proof

Study-note section SE10

Nonfunctional requirements include quality attributes and constraints. Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning. Security concerns protection; compatibility concerns cooperating environments; portability concerns moving environments; maintainability/supportability concern change, diagnosis and evolution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 78 | lines 12-30

3.3.7. Identifying Non-Functional Requirements
Nonfunctional requirements describe user-visible aspects of the system that are not directly related
to the functional behavior of the system. Nonfunctional requirements span a number of issues, from user interface look and feel to response time requirements to security issues. Nonfunctional
requirements are defined at the same time as functional requirements are, because they have as much impact on the development and cost of the system.
For example, consider a mosaic display that an air traffic controller uses to track planes. A mosaic
display system complies data from a series of radar and databases (hence the term “mosaic”) into
a summary display indicating all aircraft in a certain area, including their identification, speed, and
altitude. The number of aircraft such a system can display constrains the performance of the air traffic controller and the cost of the system. If the system can only handle a few aircraft simultaneously, the system cannot be used at busy airports. On the other hand, a system able to handle a large number of aircraft is more costly and more complex to build.
Nonfunctional requirements can be elicited by investigating the following issues.
• User interface and human factors. What kind of interface should the system provide? What is the level of expertise of the users?
• Documentation. What level of document is required? Should only user documentation be provided? Should there be technical documentation for maintainers? Should the development process be documented?

Software Engineering module | PDF page 66 | lines 14-18

3.2.2. Non-Functional and Pseudo-Requirements
Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Nonfunctional requirements include quantitative constraints, such as response time (i.e., how fast the system reacts to user commands)
or accuracy (i.e., how precise are the system’s numerical answers). The following are the

Question sources

Computer Science Exit Exam · Sene 2016 · Question 30-3

Computer Science Exit Exam practice collection · Question 18

Question 132 | SM0294

In a classical waterfall model, which phase precedes the design phase?

A. Requirements analysis and specification

B. Maintenance

C. Feasibility study

D. Coding and unit testing

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A: Requirements analysis and specification

The sequential waterfall establishes the requirements specification before deriving the software design from the SRS.

Why each choice is right or wrong

A - Correct: The immediate input and preceding phase. See PDF p. 15, 18, 26.

B - Incorrect: Occurs after delivery and operation. See PDF p. 15, 18, 26.

C - Incorrect: A prior viability activity, not the immediate specification input to design. See PDF p. 15, 18, 26.

D - Incorrect: Implements a design after it has been produced. See PDF p. 15, 18, 26.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 15 | lines 16-20

> non-iterative (or sequential) process in which progress is seen as flowing steadily downwards (like
> a waterfall) through the development phases. Thus the waterfall model maintains that one should
> move to a phase only when its preceding phase is reviewed and verified. That is, each phase must
> be completed before the next phase can begin. In waterfall model phases do not overlap. But

there
are various modified waterfall models, and hence, may include slight or major modifications.

Software Engineering module | PDF page 18 | lines 13-20

System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
Software design involves identifying and describing the fundamental software system abstractions
and their relationships. The goal of the design phase is to transform the requirements specified in
the SRS document into a structure that is suitable for implementation in some programming language. In technical terms, during the design phase the software architecture is derived from the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Software Engineering module | PDF page 26 | lines 10-17

3. Requirement's specification: Requirements specification is the activity of translating the information gathered during the analysis activity in to a document that defines a set of requirements. Two types of requirements may be included in this document. User requirements are
abstract statements of the system requirements for the customer and end-user of the system; system
requirements are a more detailed description of the functionality to be provided.
4. Requirement's validation: This activity checks the requirements for realism, consistency, and completeness. During this process, errors in the requirements document are inevitably discovered.
It must then be modified to correct these problems.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 31-1

Computer Science Exit Exam practice collection · Question 19

Question 133 | SM0300

When to use the Spiral Model?

A. Where business modeling is clearly understood and in use.

B. When it is too risky to develop the overall system as one block.

C. When requirements are understood in the early stages but can change during the project's life cycle.

D. When requirements are unstable and constantly changing.

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: When major risks make whole-system development in one block unsafe.

A spiral cycle identifies objectives, assesses and reduces important risks, develops an appropriate result and plans further commitment. B directly supplies the risk-based reason for choosing that process. Clear business understanding, potentially changing requirements and continuously unstable requirements do not alone distinguish spiral from other staged or adaptive models. The explicit risk is the decisive feature.

Why each choice is right or wrong

A - Incorrect: Clear business modeling does not by itself create a reason for risk-driven spiral development.

B - Correct: Risk makes staged assessment and commitment preferable to building the whole system in one block.

C - Incorrect: Requirements that may change fit several processes and do not specifically identify spiral.

D - Incorrect: Unstable requirements alone do not distinguish spiral without its explicit risk-analysis need.

Primary note proof

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 96 | lines 12-21

> 4.7. Risk Management
> • Risk management is concerned with identifying risks and drawing up plans to minimise their effect on a project.
> • A risk is a probability that some adverse circumstance will occur
> - Project risks affect schedule or resources;
> - Product risks affect the quality or performance of the software being developed;
> - Business risks affect the organisation developing or procuring the software.
> Risk Management Process
> - Risk identification- Identify project, product and business risks;
> - Risk analysis -Assess the likelihood and consequences of these risks;

Software Engineering module | PDF page 20 | lines 18-22

the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
In principle, the waterfall model should only be used when the requirements are well understood
and unlikely to change radically during system development. However, the waterfall model

Question sources

Computer Science Exit Exam · Sene 2016 · Question 32-3

Question 134 | SS096

The process that generates activities such as project schedule, cost estimations and work breakdown structures is described as:

A. Project Executing

B. Project Initiating

C. Project Closing

D. Project Planning

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer D: Project Planning

The project plan explicitly sets out resources, work breakdown and schedule and concerns budget; these are outputs of project planning.

Why each choice is right or wrong

A - Incorrect: Execution carries out planned tasks. See PDF p. 90, 95.

B - Incorrect: Initiation frames a project but does not best identify this full planning output set. See PDF p. 90, 95.

C - Incorrect: Closing concerns completion/acceptance, not scheduling future work. See PDF p. 90, 95.

D - Correct: Directly corresponds to the listed plan contents. See PDF p. 90, 95.

Primary note proof

Study-note section SE28

Planning is continuous: identify scope/deliverables, decompose work, estimate effort/duration/cost, assign resources, sequence dependencies and revise as evidence changes. A WBS is hierarchical and covers project scope without adding unrelated work. Tools help coordinate ownership, dependencies and progress; they do not replace judgment or automatically fix bugs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 90 | lines 7-24

4.2. Project Planning
• Probably the most time-consuming project management activity.
• Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
• Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
The project plan sets out:
• The resources available to the project;
• The work breakdown;
• A schedule for the work.
Project Plan Structure
• Introduction.
• Project organisation.
• Risk analysis.
• Hardware and software resource requirements.
• Work breakdown.
• Project schedule.
• Monitoring and reporting mechanisms.

Software Engineering module | PDF page 95 | lines 3-12

4.6. Scheduling, Organization and Team Structures
Project-task scheduling is an important project planning activity. It involves deciding which tasks would
be taken up when. In order to schedule the project activities, a software project manager needs to do
the following:
• Identify all the tasks needed to complete the project and Break down large tasks into small activities.
• Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
• Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Computer Science model exam collection · Question 3

Question 135 | SS103

Which slogan is against the agile method philosophy?

A. Working software over comprehensive documentation

B. Following a plan over responding to changes

C. Individuals and interactions over processes and tools

D. Customer collaboration over contract negotiation

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Following a plan over responding to changes

Agile values working software, people interacting, customer collaboration and responsiveness to change over the respective items on the other side of each comparison. It still recognizes value in documentation, tools, contracts and plans. The reversed preference for following a plan over responding to changes conflicts with this ordering. The other three alternatives retain the Agile ordering and therefore are not slogans against its philosophy.

Why each choice is right or wrong

A - Incorrect: This preserves the Agile preference for working software while still valuing documentation.

B - Correct: This reverses Agile’s preference for responding to change over following a plan.

C - Incorrect: This preserves the preference for individuals and interactions.

D - Incorrect: This preserves the preference for customer collaboration.

Primary note proof

Study-note section SE05

> Agile favors people collaborating, useful working results, customer participation and adaptation over rigid adherence to procedures, exhaustive documents, contracts or fixed plans. The latter still have value. Agile does not mean no planning or no documentation. A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

> Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to

follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Question sources

Computer Science model exam collection · Question 10

Question 136 | SS105

What would be the most likely consequence if every software stakeholder were rushing to be approved for their requirements without discussing with
other stakeholders?

A. Results in excess requirement

B. Attains the project schedule

C. Requirements become unambiguous

D. Leads to conflicting requirement

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer D: Leads to conflicting requirement

JAD brings stakeholders together to negotiate a mutually acceptable specification; without discussion, incompatible local demands can remain unresolved.

Why each choice is right or wrong

A - Incorrect: Excess scope is possible but not the most direct consequence of unnegotiated differences. See PDF p. 81, 68.

B - Incorrect: Rushing approvals does not prove the schedule is met. See PDF p. 81, 68.

C - Incorrect: Clarity is not obtained automatically by separate approvals. See PDF p. 81, 68.

D - Correct: Competing viewpoints can yield contradictions, precisely what consensus and consistency checks address. See PDF p. 81, 68.

Primary note proof

Study-note section SE11

Correct requirements reflect actual stakeholder needs. Complete requirements address the agreed scope, normal and exceptional behavior. Consistency excludes contradictions. Clarity/unambiguity gives one intended interpretation; feasibility/realism means achievable within constraints. Verifiability means objective evidence can check fulfillment. Replace “fast for most users” with a defined metric, workload and threshold.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 81 | lines 11-21

3.4.2. Negotiating specifications with clients: Joint Application Design
Joint Application Design (JAD) is effectiveness lies in that the requirements elicitation work is done in one single workshop session in which all stakeholders participate. Users, clients, developers, and a trained session leader sit together in one room to present their viewpoint, listen
to other viewpoints, negotiate, and agree on a mutually acceptable solution. The outcome of the workshop, the final JAD document, is a complete system specification document that includes definitions of data elements, work flows, and interface screens. Because the final document is jointly developed by the stakeholders (that is, the participants who not only have an interest in the
success of the project, but also can make substantial decisions) the final JAD document represents
an agreement between users, clients, and developers, and thus minimizes requirements changes later in the development process.

Software Engineering module | PDF page 68 | lines 18-26

system specification. Requirement validation involves checking if the specification is correct, complete, consistent, unambiguous, and realistic. A specification is correct if it represents the client's view of the system (i.e., everything in the requirements model accurately represents an aspect of the system). It is complete if all possible scenarios through the system are described, including exceptional behavior (i.e., all aspects of the system are represented in the requirements
model). The system specification is consistent if it does not contradict itself. The system specification is unambiguous if exactly one system is defined (i.e., it is not possible to interpret the specification two or more different ways). Finally, it is realistic if the system can be implemented within constraints. These properties are illustrated with UML instance diagrams in

Question sources

Computer Science model exam collection · Question 12

Question 137 | SS113

If the estimate for total discounted benefits for a project is Birr 120,000 and total discounted cost is Birr 100,000, what is the estimated return on
investment (ROI) percentage?

A. 30%

B. 12%

C. 10%

D. 20%

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer D: 20%.

The benefits and costs are already discounted to a common date, so use SE31 directly: ROI=(120000−100000)/100000×100%. Net benefit is Birr 20000, and dividing by Birr 100000 gives 0.20, or 20%. Dividing total benefits by costs would report a benefit-cost ratio, not the net return percentage requested.

Why each choice is right or wrong

A - Incorrect: Thirty percent does not follow from the given net benefit and cost.

B - Incorrect: Twelve percent is not 20000 divided by 100000 as a percentage.

C - Incorrect: Ten percent understates the calculated net return of twenty percent.

D - Correct: Net benefit is Birr 20000, which is 20% of the Birr 100000 cost.

Primary note proof

Study-note section SE31

> With benefits and costs already discounted to a common date, ROI=(benefits-costs)/costs*100%. Birr120000 benefits and Birr100000 costs give 20%, not 120%. Identify project risks to resources/schedule, product risks to quality, and business risks to the organization. Analyze likelihood and impact, plan responses and monitor.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 26 | lines 3-5

> effective from a business point of view and if it can be developed within existing budgetary constraints. A feasibility study should be relatively cheap and quick. The result should inform the decision of whether or not to go ahead with a more detailed analysis.

Software Engineering module | PDF page 90 | lines 7-24

> 4.2. Project Planning
> • Probably the most time-consuming project management activity.
> • Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
> • Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
> The project plan sets out:
> • The resources available to the project;
> • The work breakdown;
> • A schedule for the work.
> Project Plan Structure
> • Introduction.

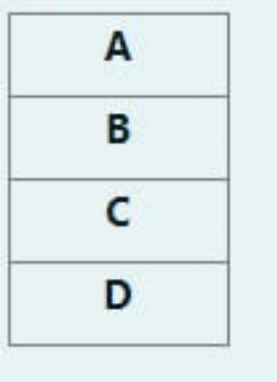
- Project organisation.
- Risk analysis.
- Hardware and software resource requirements.
- Work breakdown.
- Project schedule.
- Monitoring and reporting mechanisms.

Question sources

Computer Science model exam collection · Question 20

Question 138 | SS114

In a closed layered architecture with A,B,C,D from top to bottom, a pair (X,Y) means X directly calls Y. Calls may go only to the immediately lower layer. Which pair obeys this policy?



A. (A,C)

B. (D,A)

C. (A,B)

D. (C,B)

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: (A,B)

Closed layering permits a layer to call only the immediately lower layer. A→B follows that adjacency and direction, so C is valid. A→C skips B and violates the closed policy even though it would be possible in a suitably open architecture. D→A and C→B point upward, which the stipulated dependency policy also forbids. The ordered pair is interpreted as caller then callee.

Why each choice is right or wrong

A - Incorrect: A→C skips B.

B - Incorrect: D→A points upward.

C - Correct: A→B calls the immediately lower layer.

D - Incorrect: C→B points upward.

Primary note proof

Study-note section SE23

A layer is a logical abstraction level; a tier is a deployment separation. A closed layered architecture permits calls only to the immediately lower layer; an open one permits approved skipping. For top-to-bottom A,B,C,D, A→B obeys closed layering, A→C skips, and lower-to-upper calls violate the simple dependency rule unless explicitly mediated by a defined mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 121 | lines 6-15

6.1.4. Software architecture
As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept
of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
In this section, we describe a few sample architectures that can be used for different systems. This
is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
representative examples and refer the reader to the literature for more details.

Question sources

Computer Science model exam collection · Question 21

Question 139 | SS124

Which listed roles should document a fault if they discover it during development? Select all that apply.

A. Tester

B. Requirement engineer

C. Developer

D. Scrum master

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer A, B, C, D: Tester; Requirement engineer; Developer; Scrum master

Anyone discovering a fault should record it with enough evidence to reproduce and assess it. Testers often coordinate reporting, but requirements engineers, developers and Scrum masters should also document faults they discover. The report should identify the relevant environment, steps and expected versus actual behavior. Therefore all four roles meet the stated discoverer-based responsibility rather than only the dedicated tester.

Why each choice is right or wrong

A - Correct: A tester discovering a fault should record it.

B - Correct: A requirements engineer discovering a fault should record it.

C - Correct: A developer discovering a fault should record it.

D - Correct: A Scrum master discovering a fault should record it.

Primary note proof

Study-note section SE39

> Plan objectives, scope, resources, schedule, risks and entry/exit criteria. Analyze the test basis for testable features and defects; design cases/expected results; implement data, scripts and environment; execute and compare; report defects; evaluate completion. Analysis asks what to test and assesses testability; design asks how. Monitoring/control run across these activities.

Study-note section SE39

> Anyone finding a fault should record it; testers often coordinate reporting, but it is not their exclusive permission.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 142 | lines 24-33

> A review is the manual inspection of parts or all aspects of the system without actually executing the system. There are two types of reviews: walkthrough and inspection. In a code walkthrough, the developer informally presents the API, the code, and associated documentation of the component to the review team. The review team makes comments on the mapping of the analysis
> and object design to the code using use cases and scenarios from the analysis phase. An inspection
> is similar to a walkthrough, but the presentation of the unit is formal. In fact, in a code inspection,
> the developer is not allowed to present the artifacts (models, code, and documentation). This is done by the review team, which is responsible for checking the interface and code of the component against the requirements. It also checks the algorithms for efficiency with respect to the nonfunctional requirements. Finally, it checks comments about the code and compares them

Software Engineering module | PDF page 140 | lines 27-32

7.1 OVERVIEW OF SOFTWARE QUALITY ASSURANCE
We define testing as the systematic attempt to find errors in a planned way in the implemented software. Contrast this definition with another commonly used definition that says that “testing is
the process of demonstrating that errors are not present.” The distinction between these two definitions is important. Our definition does not mean we simply demonstrate that the program does what it is intended to do. The explicit goal of testing is to demonstrate the presence of faults.

Question sources

Computer Science model exam collection · Question 31

Question 140 | SS127

Select the one that is not correct related to generalization in system modeling.

A. Ingeneralization, the attributes and operations associated with higher-level classes are also associated with the lower-level classes

B. Facilitate easy modification of data

C. The higher-level classes are more specific than the lower-level classes by adding specific attributes and operations

D. Common information will be maintained in one place

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: The higher-level classes are more specific than the lower-level classes by adding specific attributes and operations

Generalization puts common features in more general superclasses; subclasses add or override details and are more specialized. C reverses the direction.

Why each choice is right or wrong

A - Incorrect: Inherited attributes and methods are available on subclasses. See PDF p. 104.

B - Incorrect: Common definitions can reduce duplicated modification; the wording is broad. See PDF p. 104.

C - Correct: Higher-level classes are more general, not more specific. See PDF p. 104.

D - Incorrect: Consolidating common features in a superclass removes redundancy. See PDF p. 104.

Primary note proof

Study-note section SE18

Association connects related instances. A dashed dependency arrow means a client uses a supplier. Generalization uses a solid line and hollow triangle pointing to the superclass/general

element. Shared aggregation uses a hollow diamond at the whole end and weak whole-part meaning; it does not by itself define a detailed lifecycle policy.

Study-note section SE-S02

Generalization places common concepts in a more general superclass and specialized concepts in subclasses. Available inherited attributes and operations are shared by subclasses, which may extend or override behavior. Defining common features once can reduce duplicated maintenance. The top of the hierarchy is more general, not more specialized.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 104 | lines 6-17

5.1.4. Generalization
Generalization enables us to organize concepts in to hierarchies. There may be any number of intermediate levels in between, covering more-or-less generalized concepts (e.g., Low Priority Incident, Emergency, Disaster). Such hierarchies allow us to refer to many concepts precisely. When
we use the term Incident, we mean all instances of all types of Incidents. When we use the term Emergency, we only refer to an Incident that requires an immediate response. This view of generalization stems from modeling.
In an object-oriented programming language, inheritance is a reusability technique. If a class Child
inherits from a class Parent, all the attributes and methods available on Parent are automatically available on Child. The class Child may add additional methods or override inherited methods, thus refining the class Parent. As in the case of generalization, the class at the top of the hierarchy
tends to be the most general one, whereas the leaves tend to be the most specialized.

Question sources

Computer Science model exam collection · Question 34

Question 141 | SS135

Which statement accurately describes the main evidential goal of software testing within quality assurance?

A. To achieve project timelines and deadlines.

B. To monitor the entire SDLC.

C. To generate software requirement and design documentation.

D. To assess conformance to specified requirements, reveal defects and increase confidence through evidence.

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer D: To assess conformance to specified requirements, reveal defects and increase confidence through evidence.

Testing evaluates behavior against required results, revealing failures and increasing confidence in conformance. It does not generally guarantee correctness over every possible input or environment. Meeting schedules and monitoring the whole lifecycle are broader management or assurance activities, while requirements and design documentation usually form part of the test basis. Option D states the supported evidential purpose without promising certainty.

Why each choice is right or wrong

A - Incorrect: Timelines are a project-management goal rather than the test evidence goal.

B - Incorrect: Whole-lifecycle monitoring is broader assurance/control work.

C - Incorrect: Requirements and design documents are typically inputs to testing.

D - Correct: Tests assess conformance, reveal defects and increase confidence.

Primary note proof

Study-note section SE34

> Quality assurance establishes and checks processes/standards; quality control evaluates work products. An SQA plan identifies standards, responsibilities, evaluations, reviews, audits and assurance records. Quality of conformance means following the design/specification; a product can conform to a poor specification. Verification asks whether specified obligations are implemented correctly; validation asks whether the product meets intended stakeholder needs. Both can use reviews, analysis or execution, so “human-only versus computer-only” is false.

Study-note section SE34

> Tests increase confidence and reveal defects but cannot generally guarantee correctness over every possible input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 140 | lines 27-32

> 7.1 OVERVIEW OF SOFTWARE QUALITY ASSURANCE
> We define testing as the systematic attempt to find errors in a planned way in the implemented software. Contrast this definition with another commonly used definition that says that “testing is
> the process of demonstrating that errors are not present.” The distinction between these two definitions is important. Our definition does not mean we simply demonstrate that the program does what it is intended to do. The explicit goal of testing is to demonstrate the presence of faults.

Software Engineering module | PDF page 140 | lines 11-18

Reliability is a measure of success with which the observed behavior of a system conforms to the specification of its behavior. Software reliability is the probability that a software system will not cause the failure of the system for a specified time under specified conditions. Failure is any deviation of the observed behavior from the specified behavior. An error means the system is in a state such that further processing by the system will lead to a failure, which causes the system to
deviate from its intended behavior. A fault, also called defect or bug, is the mechanical or algorithmic cause of an error. The goal of testing is to maximize the number of discovered faults, which then allows developers to correct them and increase the reliability of the system.

Question sources

Computer Science model exam collection · Question 42

Question 142 | SS137

What component hides the distinction or boundaries between various microservices from end-client application?

A. Alayered system

B. API gateway

C. API logging

D. API proxy

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: API gateway.

SE24 defines an API gateway as one entry point that routes or combines service calls and hides service boundaries from clients. That directly matches the stated role. Logging records activity and a layered system describes broader organization; a generic proxy label alone does not establish the multi-service routing and composition role given in the note.

Why each choice is right or wrong

A - Incorrect: Layering organizes abstraction levels rather than naming this single microservice-facing entry component.

B - Correct: An API gateway hides service boundaries behind a unified entry point and routes or combines requests.

C - Incorrect: API logging records activity rather than routing clients across hidden service boundaries.

D - Incorrect: A generic proxy forwards requests; the note names the broader multi-service entry-point role an API gateway.

Primary note proof

Study-note section SE24

An API gateway presents one entry point and routes/combines calls to services, hiding service boundaries from clients. It can centralize policies but also become a bottleneck. A component diagram shows components/interfaces/dependencies; a deployment diagram places software artifacts on nodes. Choose the style for the required interactions and qualities rather than selecting a universally best architecture.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 119 | lines 13-25

6.1.2. Services and subsystem interfaces
A subsystem is characterized by the services it provides to other subsystems. A service is a set of related operations that share a common purpose. A subsystem providing a notification service, for
example, defines operations to send notices, look up notification channels, and subscribe and unsubscribe to a channel.
The set of operations of a subsystem that are available to other subsystems form the subsystem interface. The subsystem interface, also referred to as the application programmer interface (API),
includes the name of the operations, their parameters, their types, and their return values. System
design focuses on defining the services provided by each subsystem that is, enumerating the operations, their parameters, and their high-level behavior. Object design will focus on defining the subsystem interfaces, that is, the type of the parameters and the return value of each operation.
The definition of a subsystem in terms of the services it provides helps us focus on its interface as opposed to its implementation. A good subsystem interface should provide as little information

Software Engineering module | PDF page 124 | lines 8-16

Client/server architecture
In the client/server architecture (Figure 6.9), a subsystem, the server, provides servicesto instances
of other subsystems called the clients, which are responsible for interacting with theuser. The request for a service is usually done via a remote procedure call mechanism or acommon object broker (e.g., CORBA or Java RMI). Control flow in the clients and the servers isindependent except for synchronization to manage requests or to receive results.
Figure 6.9 Client/server architecture (UML class diagram).
Clients request services from one or more servers. The server has no knowledge of the client. The client/server architecture is a generalization of the repository architecture.

Question sources

Computer Science model exam collection · Question 44

Question 143 | SS139

Which question is outside the architectural abstraction level and instead asks for every source-level execution detail?

A. Which architectural organization is best for delivering the functional requirements?

B. How the system will be distributed across a number of cores or processors

C. Which variable is assigned at every individual source-code execution step?

D. Which architectural patterns or styles to use

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Which variable is assigned at every individual source-code execution step?

Architecture addresses major organization, responsibilities, allocation and style decisions. Selecting an arrangement for required behavior, mapping work to processors and choosing patterns are therefore valid architectural concerns. Enumerating every source-level assignment on every execution step is implementation-level tracing, beyond the abstraction of architecture documentation. Option C supplies that genuine exception while preserving the question's comparison of architectural and detailed concerns.

Why each choice is right or wrong

A - Incorrect: Organization supporting required behavior is architectural.

B - Incorrect: Processor allocation is an architectural concern.

C - Correct: Every source-level execution step exceeds the architectural abstraction.

D - Incorrect: Style and pattern selection is architectural.

Primary note proof

Study-note section SE23

> A structure is the elements/relations themselves; a view represents selected aspects. Module structures show code organization, component-and-connector structures runtime interaction, and allocation structures map software to hardware, teams or environment. Processes/threads belong in concurrency views. Evaluate architectures against quality attributes and scenarios; processor allocation, decomposition and style selection can all be architectural concerns.

Study-note section SE21

> System design turns the analysis model into architecture: design goals, subsystem responsibilities, dependencies, hardware mapping, data management, access control, control flow and startup/shutdown/error policies. Detail/object design refines APIs, parameter/return types, exceptions, component choices and algorithms. A subsystem is a manageable logical collection with services and an interface; it need not be a physical replaceable artifact.

Study-note section SE21

Architecture documentation supports communication and records major decisions, not every source-level execution step.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 116 | lines 16-29

The analysis model describes the system completely from the actors' point of view and serves as the basis of communication between the client and the developers. The analysis model, however, does not contain information about the internal structure of the system, its hardware configuration,
or, more generally, how the system should be realized. System design is the first step in this direction. System design results in the following products:
• List of design goals, describing the qualities of the system that developers should optimize
• Software architecture, describing the subsystem decomposition in terms of subsystem responsibilities, dependencies among subsystems, subsystem mapping to hardware, and major policy decisions such as control flow, access control, and data storage.
The design goals are derived from the nonfunctional requirements. Design goals guide the decisions to be made by the developers especially when trade-offs are needed. The subsystem decomposition constitutes the bulk of system design. Developers divide the system into manageable pieces to deal with complexity: Each subsystem is assigned to a team and realized independently. In order for this to be possible, though, developers need to address system-wide

Software Engineering module | PDF page 121 | lines 6-15

6.1.4. Software architecture
As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept
of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
In this section, we describe a few sample architectures that can be used for different systems. This
is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
representative examples and refer the reader to the literature for more details.

Question sources

Computer Science model exam collection · Question 46

Question 144 | SS144

Select the one that is not related with the purpose of properly documenting software architectures

A. to Know how the system is organized and interoperate

B. For Stakeholder communication

C. For critical system design decisions

D. To easily figure out the source code flow

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: To easily figure out the source code flow

The SDD documents goals, subsystem organization, mapping and interfaces to support teams and architecture decisions. Detailed source-code control flow is a lower-level implementation concern than the document's architectural control-flow mechanisms.

Why each choice is right or wrong

A - Incorrect: Subsystem decomposition and interfaces expose organization/interoperation. See PDF p. 130.

B - Incorrect: Defines interfaces between development teams and supports communication. See PDF p. 130.

C - Incorrect: Serves as reference when architecture-level decisions are revisited. See PDF p. 130.

D - Correct: Detailed source-level flow is not the purpose of an architecture description; global control mechanisms may still be documented. See PDF p. 130.

Primary note proof

Study-note section SE21

> System design turns the analysis model into architecture: design goals, subsystem responsibilities, dependencies, hardware mapping, data management, access control, control flow and startup/shutdown/error policies. Detail/object design refines APIs, parameter/return types, exceptions, component choices and algorithms. A subsystem is a manageable logical collection with services and an interface; it need not be a physical replaceable artifact.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 130 | lines 3-11

> 6.3. DOCUMENTING SYSTEM DESIGN
> System design is documented in the System Design Document (SDD). It describes design goals set by the project, the subsystem decomposition (with UML class diagrams), the hardware/software mapping (with UML deployment diagrams), the data management, the access

control, control flow mechanisms, and boundary conditions. The SDD is used to define interfaces between teams of developers and as reference when architecture-level decisions need to be revisited. The audience for the SDD includes the project management, the system architects (i.e., the developers who participate in the system design), and the developers who design and implement each subsystem. The following is an example template for a SDD:

Question sources

Computer Science model exam collection · Question 51

Question 145 | SS147

Given the following fragment of code, how many tests are required for 100% decision coverage?
if width > length
then biggest_dimension = width
if height > width
then biggest_dimension = height
end_if
else biggest_dimension = length
if height > length
then biggest_dimension = height
end_if
end_if

A. ona. 4

B. 6

C. Ole 2

D. 3

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer A: Four tests for 100% decision coverage.

There are three decisions: width>length and one separate height comparison inside each outer branch. Covering both inner outcomes requires two tests with outer true and two with outer false. For example (w,l,h)=(3,2,4),(3,2,1),(2,3,4),(2,3,1) covers all outcomes. One execution cannot visit both outer branches, so two or three tests cannot suffice; six are unnecessary.

Why each choice is right or wrong

A - Correct: Four tests cover true/false of the inner decision on each of the two outer branches.

B - Incorrect: Six can cover the decisions, but four suffice; it is not the required minimum.

C - Incorrect: Two cover the outer decision but cannot cover both outcomes of both separate inner decisions.

D - Incorrect: Three leave at least one inner decision outcome uncovered because each outer branch needs two tests.

Primary note proof

Study-note section SE38

> Statement coverage is the number of distinct executable statements exercised divided by the total executable statements in the declared test scope. Unreachable statements can prevent100%; any justified exclusions must be documented with the adjusted scope. Decision coverage counts decision outcomes; branch coverage counts control-flow edges, including unconditional edges under the stated convention. Testing both true and false of each decision may need more than two overall test cases when branches nest. Coverage is evidence of exercised structure, not proof of all behavior.

Study-note section SE38

> In the width/length/height program, the outer if has separate inner decisions in each branch. Need outer true with inner true and false, and outer false with its inner true and false: four tests, for example (w,l,h)=(3,2,4),(3,2,1),(2,3,4),(2,3,1). One run cannot exercise both outer branches. All paths may be infeasible or infinite with loops; specify the chosen coverage target.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 149 | lines 24-29

> Path testing
> Path testing is a white box testing technique that identifies faults in the implementation of the component. The assumption behind path testing is that, by exercising all possible paths through the code at least once, most faults will trigger failures. The identification of paths requires knowledge of the source code and data structures.
> The starting point for path testing is the flow graph. A flow graph consists of nodes representing

Software Engineering module | PDF page 147 | lines 3-8

> • A test stub is a partial implementation of components on which the tested component depends. A test driver is a partial implementation of a component that depends on the tested component. Test stubs and drivers enable components to be isolated from the rest of the system for testing.
> • A correction is a change to a component. The purpose of a correction is to repair a fault. Note that a correction can introduce new faults.

Question sources

Computer Science model exam collection · Question 54

Question 146 | SS148

Using classical up-front waterfall planning, usable incremental additions and timeboxed adaptive Agile, which lifecycle statements are NOT true? Select all that apply.

A. Ina waterfall life cycle, the project scope, time, and cost are determined in the early phases of the life cycle

B. Inagile life cycle the project scope is generally determined early in the project life cycle, but time and cost estimates are routinely modified

C. Inanincremental life cycle, the deliverable is produced through a series of iterations that successively add functionality within a predetermined time frame

D. Inagile life cycles, the project scope cannot be outlined and agreed before the start of iteration.

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B, D: Inagile life cycle the project scope is generally determined early in the project life cycle, but time and cost estimates are routinely modified; Inagile life cycles, the project scope cannot be outlined and agreed before the start of iteration.

Classical waterfall establishes planned baselines early, and incremental work adds functionality in successive deliveries. Agile instead adapts content while its timeboxes tend to remain fixed, so B misstates the general organizing pattern. D is also false because Agile permits outlining and agreeing scope before an iteration; planning continues as evidence changes. Adaptation does not mean that scope discussion or early agreement is prohibited.

Why each choice is right or wrong

A - Incorrect: Early planned baselines fit classical waterfall.

B - Correct: The claim reverses the timeboxed/adaptive-content characterization.

C - Incorrect: Successive additions of usable functionality fit incremental delivery.

D - Correct: Agile permits scope discussion and agreement before an iteration.

Primary note proof

Study-note section SE05

> Scrum uses timeboxed Sprints of at most one month, with a Sprint Goal, selected work and a usable Increment meeting a Definition of Done. Scope is discussed before and during planning; it is not forbidden to agree scope. Timeboxes tend to stay fixed while content is adapted without endangering the goal. Agile can be combined with required assurance/documentation; safety-critical context alone cannot prove it is unsuitable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

> Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to
> follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Software Engineering module | PDF page 22 | lines 8-12

> This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
> also provide a platform for evaluation by the user.

Software Engineering module | PDF page 20 | lines 18-22

> the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
> In principle, the waterfall model should only be used when the requirements are well understood
> and unlikely to change radically during system development. However, the waterfall model

Question sources

Computer Science model exam collection · Question 55

Question 147 | SS151

Which design pattern is used to restore the state of an object to its previous state?

A. Iterator pattern

B. Observer pattern

C. Momento pattern

D. Visitor pattern

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Memento, printed as Momento.

SE27 explicitly identifies Memento as storing and restoring state while hiding representation, and notes that Momento in this question bank is a spelling error for Memento. That makes C the intended pattern name. Iterator traverses a collection and Observer notifies subscribers; neither supplies the state-snapshot restoration responsibility asked for here.

Why each choice is right or wrong

A - Incorrect: Iterator traverses elements rather than storing a restorable snapshot of object state.

B - Incorrect: Observer notifies subscribers about changes rather than restoring an earlier object state.

C - Correct: Momento refers here to Memento, the state capture-and-restoration pattern.

D - Incorrect: Visitor is not the state-snapshot restoration pattern; the note assigns that role to Memento.

Primary note proof

Study-note section SE27

> Factory Method lets subclasses choose a concrete product through a creation operation, reducing coupling to concrete constructors. A simple factory centralizes creation but is not identical to Factory Method. Singleton controls access to one instance; Observer notifies subscribers; Memento stores/restores state while hiding representation; Iterator traverses; Decorator adds behavior through wrapping. The pattern name is Memento, not Momento.

Study-note section SE27

> Variability can use selectable features, omission/inclusion and extension points; overloading supplies alternate signatures in code but is not every variability mechanism. Testability improves with controllable inputs, observable outputs, replaceable dependencies and limited structural complexity. A sandbox can isolate execution. Assess each design claim independently before evaluating a combined statement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 104 | lines 6-17

> 5.1.4. Generalization
> Generalization enables us to organize concepts in to hierarchies. There may be any number of intermediate levels in between, covering more-or-less generalized concepts (e.g., Low Priority Incident, Emergency, Disaster). Such hierarchies allow us to refer to many concepts precisely. When
> we use the term Incident, we mean all instances of all types of Incidents. When we use the term Emergency, we only refer to an Incident that requires an immediate response. This view of generalization stems from modeling.
> In an object-oriented programming language, inheritance is a reusability technique. If a class Child
> inherits from a class Parent, all the attributes and methods available on Parent are automatically available on Child. The class Child may add additional methods or override inherited methods, thus refining the class Parent. As in the case of generalization, the class at the top of the hierarchy
> tends to be the most general one, whereas the leaves tend to be the most specialized.

Software Engineering module | PDF page 121 | lines 6-15

6.1.4. Software architecture
As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept
of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
In this section, we describe a few sample architectures that can be used for different systems. This
is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
representative examples and refer the reader to the literature for more details.

Question sources

Computer Science model exam collection · Question 58

Question 148 | SS153

Which of the following shows the five stages in Tuckman's model of team development, in sequential order?

A. forming, storming, performing, norming, and adjourning

B. norming, forming, storming, performing, and adjourning

C. storming, forming, norming, performing, and adjourning

D. forming, storming, norming, performing, and adjourning

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer D: Forming → storming → norming → performing → adjourning.

SE32 lists orientation first, then conflict, development of norms, effective performance and closure. Their stage names are forming, storming, norming, performing and adjourning. D preserves that order. The other alternatives either put performance before norms or move orientation after later stages, contradicting the taught sequence.

Why each choice is right or wrong

A - Incorrect: It places performing before norming, reversing the stated development sequence.

B - Incorrect: It starts with norming before the initial forming and storming stages.

C - Incorrect: It places storming before the initial forming stage.

D - Correct: It matches the taught progression from orientation through conflict, norms, performance and closure.

Primary note proof

Study-note section SE32

Functional organizations group specialists by discipline; project organizations group people around projects. Functional specialization helps staffing and knowledge continuity but may create handoff barriers. Democratic teams share decisions; chief-programmer teams centralize technical coordination. These are different classification dimensions. Tuckman's sequence is forming→storming→norming→performing→adjourning: orientation, conflict, norms, effective work and closure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 96 | lines 3-11

Team Structure
• Team structure addresses the issue of organization of the individual project teams. There are some possible ways in which the individual project teams can be organized.
• Problems of different complexities and sizes often require different team structures for chief solution.
• The most common types of team structure are:
· Chief Programmer Team
· Democratic Team
· Mixed Control Team Organization

Question sources

Computer Science model exam collection · Question 60

Question 149 | SS154

Which one of the following is not among the purposes of software testing?

A. Requesting more design and implementation time

B. Identifying shortcomings

C. Improving product acceptance

D. Enhancing reliability

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer A: Requesting more design and implementation time

Testing detects faults, supports client acceptance and enables corrections that improve reliability. Requesting more implementation time is a possible project consequence, not a testing purpose.

Why each choice is right or wrong

A - Correct: A request for time is not the fault-detection or confidence purpose. See PDF p. 140, 140, 145.

B - Incorrect: Identifying faults/shortcomings is the explicit goal. See PDF p. 140, 140, 145.

C - Incorrect: Client acceptance is checked through acceptance tests. See PDF p. 140, 140, 145.

D - Incorrect: Fault discovery permits corrections and increased reliability. See PDF p. 140, 140, 145.

Primary note proof

Study-note section SE34

> A fault/defect is an underlying problem; an error may be an incorrect internal state; a failure is an externally observed departure from required service. Reliability is a product quality attribute; low cost or short development time are project outcomes. Tests increase confidence and reveal defects but cannot generally guarantee correctness over every possible input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 140 | lines 27-32

> 7.1 OVERVIEW OF SOFTWARE QUALITY ASSURANCE
> We define testing as the systematic attempt to find errors in a planned way in the implemented software. Contrast this definition with another commonly used definition that says that “testing is
> the process of demonstrating that errors are not present.” The distinction between these two definitions is important. Our definition does not mean we simply demonstrate that the program does what it is intended to do. The explicit goal of testing is to demonstrate the presence of faults.

Software Engineering module | PDF page 140 | lines 11-18

> Reliability is a measure of success with which the observed behavior of a system conforms to the specification of its behavior. Software reliability is the probability that a software system will not cause the failure of the system for a specified time under specified conditions. Failure is any deviation of the observed behavior from the specified behavior. An error means the system is in a state such that further processing by the system will lead to a failure, which causes the system to
> deviate from its intended behavior. A fault, also called defect or bug, is the mechanical or algorithmic cause of an error. The goal of testing is to maximize the number of discovered faults, which then allows developers to correct them and increase the reliability of the system.

Software Engineering module | PDF page 145 | lines 3-5

> o Acceptance testing and installation testing check the requirements against the
> project agreement and should be done by the client, if necessary with support by
> the developers.

Question sources

Computer Science model exam collection · Question 61

Question 150 | SS155

During which listed activity should the test plan define configuration-management procedures alongside objectives, resources and schedule, before those procedures are applied during execution?

A. During test execution

B. During test planning

C. During test closing

D. During test initiation

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: During test planning

Test planning specifies intended work, objectives, resources and schedule; configuration-management procedures should be defined early as part of that preparation. They then apply throughout testing to preserve reproducible builds, data and environments. Execution and closing use those procedures rather than being their exclusive establishment point. Initiation may set context, but the question specifically asks for the activity that defines them in the test plan.

Why each choice is right or wrong

A - Incorrect: Execution applies the planned procedures to tested versions.

B - Correct: Test planning defines procedures with the other planned work.

C - Incorrect: Closing preserves final artifacts under established control.

D - Incorrect: Initiation sets context rather than the specified test-plan definition activity.

Primary note proof

Study-note section SE33

> Configuration management identifies controlled items, versions/baselines, change procedures, status and audits. Items may include source code, requirements, manuals, test plans, environments and build configurations. A team’s ordinary browser is usually a tool rather than product CI, but a controlled test-browser version may be an environment CI.

Study-note section SE39

> A test plan describes intended work; incident/defect reports are normally execution outputs. A defect report should identify build/environment, steps, expected versus actual behavior, evidence and severity. Anyone finding a fault should record it; testers often coordinate

reporting, but it is not their exclusive permission. Configuration control makes the tested version reproducible.

Study-note section SE33

Plan test configuration management early and apply it throughout testing, not at just one exclusive phase.

Study-note section SE39

Analysis asks what to test and assesses testability; design asks how.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 142 | lines 5-10

Configuration management avoids faults caused by undisciplined change in the system models. For example, it is a common mistake to a change a subsystem interface without notifying all developers of calling components. Configuration management can make sure that, if analysis models and code are becoming inconsistent with one another, analysts and implementor are notified. The assumption behind configuration management is that the system contains many less
faults if change is controlled.

Software Engineering module | PDF page 90 | lines 7-24

4.2. Project Planning
• Probably the most time-consuming project management activity.
• Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
• Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
The project plan sets out:
• The resources available to the project;
• The work breakdown;
• A schedule for the work.
Project Plan Structure
• Introduction.
• Project organisation.
• Risk analysis.
• Hardware and software resource requirements.
• Work breakdown.
• Project schedule.
• Monitoring and reporting mechanisms.

Question sources

Computer Science model exam collection · Question 62

Question 151 | SS156

"The system architecture should be designed using fine-grain, self-contained components. Producers of data should be separated from consumers
and shared data structures should be avoided". For which requirement this architecture description applies?

A. Performance

B. Availability

C. Maintainability

D. Security

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Maintainability

Independent self-contained subsystems and avoiding shared attribute data reduce coupling, limiting the impact of future changes. That directly serves maintainability.

Why each choice is right or wrong

A - Incorrect: Fine-grained separation can carry overhead and is not primarily the stated speed optimization. See PDF p. 120, 118.

B - Incorrect: Availability requires continuity/failure treatment beyond component separation alone. See PDF p. 120, 118.

C - Correct: Matches minimizing change impact through low coupling. See PDF p. 120, 118.

D - Incorrect: Security needs explicit access-control policy; independence alone does not establish it. See PDF p. 120, 118.

Primary note proof

Study-note section SE43

> Choose designs by their effect on stated goals. Fine-grained self-contained components, separation of producers/consumers and limited shared mutable data often improve maintainability. Trade-offs remain: interfaces can add latency, isolation can complicate transactions, and global state can simplify one operation while coupling many others. Prefer measurable quality scenarios to counting components alone.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 120 | lines 6-12

> 6.1.3. Coupling and coherence
> Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will

have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Software Engineering module | PDF page 118 | lines 11-16

6.1.1. Subsystems and classes
In order to reduce the complexity of the application domain, we identified smaller parts called classes and organized them into packages. Similarly, to reduce the complexity of the solution domain, we decompose a system into simpler parts, called subsystems, which are made of a number of solution domain classes. In the case of complex subsystems, we recursively apply this principle and decompose a subsystem into simpler subsystems (see Figure 6.2).

Question sources

Computer Science model exam collection · Question 63

Question 152 | SS159

Which of the following is a process of selecting program paths in such a manner that certain branches (i.e., outgoing edges of nodes) of a control flow
graph are covered by the execution of those paths?

A. Branch boundary

B. Branch coverage

C. Branch control

D. Branch expansion

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: Branch coverage.

The stem explicitly defines branches as outgoing control-flow edges. SE38 identifies branch coverage as counting exercised control-flow edges under the declared convention. This differs from merely counting executed statements or evaluating decision outcomes. Branch boundary, branch control and branch expansion are not the coverage criterion described by selecting paths that exercise graph edges.

Why each choice is right or wrong

A - Incorrect: Branch boundary is not the defined edge-coverage criterion in the note.

B - Correct: Branch coverage measures whether control-flow edges are exercised by the selected test paths.

C - Incorrect: Branch control is not the name of the specified structural coverage measure.

D - Incorrect: Branch expansion does not name the outgoing-edge coverage criterion described.

Primary note proof

Study-note section SE38

> Statement coverage is the number of distinct executable statements exercised divided by the total executable statements in the declared test scope. Unreachable statements can prevent100%; any justified exclusions must be documented with the adjusted scope. Decision coverage counts decision outcomes; branch coverage counts control-flow edges, including unconditional edges under the stated convention. Testing both true and false of each decision may need more than two overall test cases when branches nest. Coverage is evidence of exercised structure, not proof of all behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 149 | lines 24-29

> Path testing
> Path testing is a white box testing technique that identifies faults in the implementation of the component. The assumption behind path testing is that, by exercising all possible paths through the code at least once, most faults will trigger failures. The identification of paths requires knowledge of the source code and data structures.
> The starting point for path testing is the flow graph. A flow graph consists of nodes representing

Question sources

Computer Science model exam collection · Question 66

Question 153 | SS161

Which of the following is the correct sequence that will be followed during software evolution process?

A. Change request, release planning, change implementation, impact analysis, system release

B. Change request, impact analysis, release planning, change implementation, system release

C. Impact analysis, Change request, change implementation, release planning, system release

D. Change request, impact analysis, change implementation, release planning, system release

Hint

Classify the motive for the change before its implementation: repair, environment, useful capability or reduced future risk.

Answer B: Change request → impact analysis → release planning → implementation → release.

SE40 gives the sequence directly. A requested change is analyzed before commitment to a release, then planned, implemented and tested, and finally released. B preserves those dependencies. The alternatives analyze consequences too late, begin analysis before a request exists, or implement before release planning. Implementation here includes the testing work omitted from the abbreviated options.

Why each choice is right or wrong

A - Incorrect: It postpones impact analysis until after implementation, too late to guide planning and commitment.

B - Correct: It preserves request, analysis, planning, implementation/testing and release in the taught order.

C - Incorrect: It begins impact analysis before identifying a change request and implements before release planning.

D - Incorrect: It places implementation before release planning instead of planning the analyzed change first.

Primary note proof

Study-note section SE40

> A typical evolution sequence is change request→impact analysis→release planning→implementation/testing→release. Among a list lacking the request step, impact analysis may be the earliest listed technical activity. Major restructuring is reengineering; if the architecture itself changes, architectural transformation is the more specific description. Markets, technology, stakeholders and funding can all change scope or priorities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 41 | lines 16-17

> ✓ Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous and the software must evolve to remain useful.

Software Engineering module | PDF page 19 | lines 24-29

> 5. Operation and maintenance: Normally (although not necessarily), this is the longest life cycle phase. Maintenance of a typical software product requires much more than the effort necessary to
> develop the product itself. The system is installed and put into practical use. Maintenance involves
> correcting errors which were not discovered in earlier stages of the life cycle, improving the implementation of system units and enhancing the system ‘s services as new requirements are discovered.

Question sources

Computer Science model exam collection · Question 68

Question 154 | SS162

Which of the following UML element is wrongly applied?

A. Use case diagram represents an interaction with the system

B. Sequence diagram shows the sequence of interactions required to complete some operation

C. An activity diagram primarily specifies the persistent lifecycle states of one object and the event-triggered transitions between those states.

D. Class diagram shows the object classes in a system and their relation

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: An activity diagram primarily specifies the persistent lifecycle states of one object and the event-triggered transitions between those states.

Use-case diagrams summarize actor goals, sequence diagrams order interactions across participants, and class diagrams describe static types and relationships. An activity diagram models workflow and action/control flow. Persistent lifecycle states and event-triggered transitions of a particular object instead call for a state machine. The revised third statement makes that mismatch explicit; the original generic event wording was too broad because activities can also respond to events.

Why each choice is right or wrong

A - Incorrect: Use-case diagrams summarize external roles and system goals/interactions.

B - Incorrect: Sequence diagrams show ordered participant interactions.

C - Correct: The specified lifecycle-state behavior belongs to a state machine rather than an activity workflow.

D - Incorrect: Class diagrams describe static classes and their relationships.

Primary note proof

Study-note section SE20

> An activity diagram models workflow: filled circle start, rounded actions, guarded decision diamond, thick fork/join bars and partitions/swimlanes for responsibility. A diamond can also merge alternative flows; a join synchronizes concurrent flows. A state machine models states and transitions triggered by events, often with guards/actions, for one object's lifecycle.

Study-note section SE19

> Sequence diagrams show interactions across lifelines, with order/time generally downward. Messages cross between participants; activation rectangles show execution. A creation message starts a lifeline; an X marks modeled destruction, not a guarantee about when a garbage

collector runs. Synchronous calls wait for completion; asynchronous messages permit the sender to proceed.

Study-note section SE17

A class box lists name, attributes and operations; + public, - private, # protected and ~ package indicate visibility. A class diagram shows static types and relationships, while an object diagram shows particular instances/links.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 47 | lines 30-39

Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
Use cases describe the behavior of the system as seen from an actor's point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate a
use case to access the system functionality. The use case can then initiate other use cases and gather
more information from the actors. When actors and use cases exchange information, they are said
to communicate. We will see later that we represent these exchanges with communication relationships.

Software Engineering module | PDF page 54 | lines 26-35

both buttons simultaneously. Figure 2.10 depicts a sequence diagram for an actor setting his 2Bwatch one minute ahead.
Each column represents an object that participates in the interaction. The vertical axis represents
time from top to bottom. Messages are shown by arrows. Labels on arrows represent message names and may contain arguments. Activations (i.e., executing methods) are depicted by vertical rectangles. Actors are shown as the leftmost column.
Sequence diagrams can be used to describe either an abstract sequence (i.e., all possible interactions) or concrete sequences (i.e., one possible interaction, as in Figure 2.10). When describing all possible interactions, sequence diagrams also provide notations for conditionals and
iterators.

Software Engineering module | PDF page 56 | lines 18-24

2.3.4. Activity diagrams
The outgoing transitions are triggered by the completion of an action associated with the state. This is called an action state. By convention, the name of a state denotes a condition, whereas the name of an action state denotes an action. Activity diagrams are statechart diagrams whose states
are action states. is an activity diagram corresponding to the state diagram. An alternate and

equivalent view of activity diagrams is to interpret action states as control flow between activities
and transitions; that is, the arrows are interpreted as sequential constraints between activities.

Software Engineering module | PDF page 55 | lines 4-9

2.3.3. State chart Diagram
A UML statechart is a notation for describing the sequence of states an object goes through in response to external events. Statecharts are extensions of the finite state machine model. On the one hand, statecharts provide notation for nesting states and state machines (i.e., a state can be described by a state machine). On the other hand, statecharts provide notation for binding transitions with message sends and conditions on objects.

Software Engineering module | PDF page 51 | lines 14-24

2.3.2. Class Diagrams
Classes and objects
Class diagrams describe the structure of the system in terms of classes and objects. Classes are abstractions that specify the attributes and behavior of a set of objects. Objects are entities that encapsulate state and behavior. Each object has an identity: It can be referred individually and is distinguishable from other objects.
In UML, classes and objects are depicted by boxes including three compartments. The top compartment displays the name of the class or object. The center compartment displays its attributes, and the bottom compartment displays its operations. The attribute and operation compartment can be omitted for clarity. Object names are underlined to indicate that they are instances. By convention, class names start with an uppercase letter. Objects in object diagrams

Question sources

Computer Science model exam collection · Question 69

Question 155 | SS163

Select the wrong statement regarding traceability in software development projects.

A. Traceability shows the dependency among two or more requirements if any

B. Requirements should be linked to the respective stakeholder who generated them

D. Every design element(component) should be linked back to the requirement

G. Traceability traces the overall cost and schedule of every activity in the requirement

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer G: Traceability traces the overall cost and schedule of every activity in the requirement

Traceability records where requirements came from, how requirements depend on one another, and how each requirement is realized and checked in design, code and tests. Source links support rationale, and forward or backward links support change-impact analysis. A cost-and-schedule ledger instead belongs to project planning and control. It can reference requirements, but recording every activity's cost and schedule is not the definition of requirements traceability.

Why each choice is right or wrong

A - Incorrect: Dependency links among requirements are a valid traceability relation.

B - Incorrect: Source or stakeholder links record a requirement's provenance.

D - Incorrect: Design-to-requirement links support realization and impact analysis.

G - Correct: An overall activity cost-and-schedule ledger is project control, not the defining traceability relation.

Primary note proof

Study-note section SE11

> Trace each requirement backward to its source/rationale and forward to design, code and tests. Record dependencies among requirements. A traceability matrix links these artifacts and helps impact analysis; it is not just a cost/schedule ledger. Stakeholders must negotiate conflicting requirements rather than independently approving incompatible demands. Scope, understanding and volatility are distinct elicitation difficulties.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 70 | lines 3-16

> 3.2.5. Verifiability and Traceability
> Two more desirable properties of a system specification are that it be verifiable and traceable. The
> specification is verifiable if, once the system is built; a repeatable test can be designed to demonstrate that the system fulfills the requirement. For example, a mean time to failure of a hundred years for SatWatch would be difficult to achieve (assuming it is realistic in the first place).
> The following requirements are additional examples of non-verifiable requirements:
> • The product shall have a good user interface (good is not defined).
> • The product shall be error free (requires large amount of resources to establish).
> • The product shall respond to the user with 1 second for most cases ("most cases" is not defined).
> A system specification is traceable if each system function can be traced to its corresponding set of requirements. Traceability is not a constraint on the content of the specification, but rather, on its organization. Traceability facilitates the development of tests and the systematic validation of the design against the requirements.

Software Engineering module | PDF page 90 | lines 7-24

4.2. Project Planning
• Probably the most time-consuming project management activity.
• Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
• Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
The project plan sets out:
• The resources available to the project;
• The work breakdown;
• A schedule for the work.
Project Plan Structure
• Introduction.
• Project organisation.
• Risk analysis.
• Hardware and software resource requirements.
• Work breakdown.
• Project schedule.
• Monitoring and reporting mechanisms.

Question sources

Computer Science model exam collection · Question 70

Question 156 | SS164

Which named prioritization heuristic suggests that a small subset of causes may account for much of the observed impact, often summarized as 80/20?

A. Pareto principle

B. Pairwise principle

C. Partition principle

D. Parametric principle

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer A: Pareto principle

Pareto's 80/20 idea is a prioritization heuristic: a relatively small set of causes may account for a large share of observed impact. It encourages examining evidence to focus improvement effort. It does not promise that exactly one fifth of the effort fixes exactly four fifths of all defects. The revised question asks for this named heuristic, so the other principle names do not identify the rule described.

Why each choice is right or wrong

A - Correct: Pareto names the small-subset/large-impact heuristic often expressed as 80/20.

B - Incorrect: This is not the name of the described 80/20 prioritization heuristic.

C - Incorrect: Partitioning does not name the described concentration-of-impact heuristic.

D - Incorrect: Parametric reasoning does not name the described concentration-of-impact heuristic.

Primary note proof

Study-note section SE43

> Pareto’s 80/20 rule is a prioritization heuristic: a small subset of causes may account for much of the observed impact. It does not guarantee 80% of defects can be fixed with exactly 20% effort. Refactor to preserve behavior while improving structure; optimize only against measured needs; record rationale and update traceability and regression evidence after change.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 91 | lines 37-42

> • Software Metrics can be used to indicate the basic attributes of the software. The indicative metrics, not exhaustive, are
> • Size - Line of code ,Function point count
> • Quality- Reliability, Accuracy, Dependability, and these are measured through some measures like errors, failures and number of runs between failures ,etc.
> • Execution Time -process time of execution of a critical process and many more.

Question sources

Computer Science model exam collection · Question 71

Question 157 | SS166

A business expects changing requirements and organizes development around repeated customer collaboration and incremental replanning; risk-analysis cycles are not its organizing priority. Which offered approach most directly matches these priorities?

A. Code and run

B. Spiral model

C. Agile development method

D. Water fall model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Agile development method

The scenario's organizing emphasis is repeated customer collaboration and adaptive planning as needs change, which identifies Agile. Spiral can also accommodate change, but organizes its cycles around explicit risk assessment and reduction. Classical waterfall relies more heavily on stable requirements and planned phase commitments. Code-and-run lacks the described disciplined planning and feedback organization. Changing requirements alone would not uniquely determine a process, so the added priorities matter.

Why each choice is right or wrong

A - Incorrect: Code-and-run does not supply the disciplined adaptive planning and feedback organization stated.

B - Incorrect: Spiral is specifically organized around risk assessment and reduction rather than the stated primary emphasis.

C - Correct: Agile makes customer collaboration and adaptation central to planning.

D - Incorrect: Classical waterfall is a poorer fit for repeated replanning of evolving needs.

Primary note proof

Study-note section SE05

> Agile favors people collaborating, useful working results, customer participation and adaptation over rigid adherence to procedures, exhaustive documents, contracts or fixed plans. The latter still have value. Agile does not mean no planning or no documentation. A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

> Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to
> follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Software Engineering module | PDF page 20 | lines 18-22

> the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to

respond to changing customer requirements.
In principle, the waterfall model should only be used when the requirements are well understood
and unlikely to change radically during system development. However, the waterfall model

Question sources

Computer Science model exam collection · Question 73

Question 158 | SS168

Which statement is correct about the differences between functional and non-functional requirements?

A. Functional requirements are the services the system provide while non-functional requirements are constraints on the services or functions offered by the system

B. Functional requirements are decided by the customer while non-functional requirements are decided by developers

C. Both are the same most of the time and only vary under certain contexts

D. Unlike functional requirement, nonfunctional requirements are stable

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: Functional requirements are the services the system provide while non-functional requirements are constraints on the services or functions offered by the system

Required interactions are distinct from user-visible quality constraints such as response time and accuracy; users and clients participate in defining both.

Why each choice is right or wrong

A - Correct: Correct service-versus-constraint distinction. See PDF p. 65, 66, 78.

B - Incorrect: Both types involve users/clients and developers; they are not separated by who decides. See PDF p. 65, 66, 78.

C - Incorrect: The module explicitly distinguishes their subject matter. See PDF p. 65, 66, 78.

D - Incorrect: No immunity to changing stakeholder needs is stated for nonfunctional requirements. See PDF p. 65, 66, 78.

Primary note proof

Study-note section SE10

Nonfunctional requirements include quality attributes and constraints. Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning. Security concerns protection; compatibility concerns cooperating environments; portability concerns moving environments; maintainability/supportability concern change, diagnosis and evolution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 65 | lines 19-23

3.2.1. Functional Requirements
Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. For example, the following is an example of functional requirements for SatWatch, a watch that resets itself without user intervention:

Software Engineering module | PDF page 66 | lines 14-18

3.2.2. Non-Functional and Pseudo-Requirements
Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Nonfunctional requirements include quantitative constraints, such as response time (i.e., how fast the system reacts to user commands)
or accuracy (i.e., how precise are the system's numerical answers). The following are the

Software Engineering module | PDF page 78 | lines 12-30

3.3.7. Identifying Non-Functional Requirements
Nonfunctional requirements describe user-visible aspects of the system that are not directly related
to the functional behavior of the system. Nonfunctional requirements span a number of issues, from user interface look and feel to response time requirements to security issues. Nonfunctional
requirements are defined at the same time as functional requirements are, because they have as much impact on the development and cost of the system.
For example, consider a mosaic display that an air traffic controller uses to track planes. A mosaic
display system complies data from a series of radar and databases (hence the term "mosaic") into
a summary display indicating all aircraft in a certain area, including their identification, speed, and
altitude. The number of aircraft such a system can display constrains the performance of the air traffic controller and the cost of the system. If the system can only handle a few aircraft simultaneously, the system cannot be used at busy airports. On the other hand, a system able to handle a large number of aircraft is more costly and more complex to build.
Nonfunctional requirements can be elicited by investigating the following issues.
• User interface and human factors. What kind of interface should the system provide? What is the level of expertise of the users?
• Documentation. What level of document is required? Should only user documentation be

provided? Should there be technical documentation for maintainers? Should the development process be documented?

Question sources

Computer Science model exam collection · Question 75

Question 159 | SS169

Which paired activity assesses requirements/system testability and then designs test cases and expected results?

A. Test analysis and requirements

B. Test analysis and design

C. Test analysis and planning

D. Test analysis and Implementation

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: Test analysis and design

Test analysis examines the test basis, identifies what can be tested and evaluates testability. Test design then specifies how to test it through cases and expected results. The analysis-and-design pair therefore covers both responsibilities. Requirements work, planning and implementation are relevant elsewhere, but do not replace case design: planning addresses resources and scope, while implementation prepares data, scripts and the environment.

Why each choice is right or wrong

A - Incorrect: Requirements work does not supply the requested case-design half.

B - Correct: Analysis assesses testability and design specifies cases and expected results.

C - Incorrect: Planning addresses intended work, resources and schedule rather than case design.

D - Incorrect: Implementation prepares scripts, data and environment rather than being the design half.

Primary note proof

Study-note section SE39

A test plan describes intended work; incident/defect reports are normally execution outputs. A defect report should identify build/environment, steps, expected versus actual behavior, evidence and severity. Anyone finding a fault should record it; testers often coordinate reporting, but it is not their exclusive permission. Configuration control makes the tested version reproducible.

Study-note section SE39

> Analysis asks what to test and assesses testability; design asks how.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 147 | lines 14-18

> • Unit testing, which finds faults by isolating an individual component using test stubs and drivers and by exercising the component using a test case
> • Integration testing, which finds faults by integrating several components together
> • System testing, which focuses on the complete system, its functional and nonfunctional requirements, and its target environment

Question sources

Computer Science model exam collection · Question 76

Question 160 | SS174

Which parameter is used to assess and evaluate software architectures?

A. Number of components in the architecture

B. Durability of the architecture

C. Responsiveness of the architectures

D. Architectural quality attributes

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: Architectural quality attributes

Design goals are qualities derived from nonfunctional requirements; choices are evaluated consistently against performance, dependability, maintenance, cost and end-user criteria.

Why each choice is right or wrong

A - Incorrect: Component count alone does not establish architecture quality. See PDF p. 127, 116.

B - Incorrect: Durability alone does not span the taught criteria. See PDF p. 127, 116.

C - Incorrect: Responsiveness is only one performance criterion. See PDF p. 127, 116.

D - Correct: Covers the full set of quality criteria used to judge architectural decisions. See PDF p. 127, 116.

Primary note proof

Study-note section SE23

A layer is a logical abstraction level; a tier is a deployment separation. A closed layered architecture permits calls only to the immediately lower layer; an open one permits approved skipping. For top-to-bottom A,B,C,D, A→B obeys closed layering, A→C skips, and lower-to-upper calls violate the simple dependency rule unless explicitly mediated by a defined mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 127 | lines 10-28

6.2.1. Identifying design goals
The definition of design goals is the first step of system design. It identifies the qualities that our system should focus on. Many design goals can be inferred from the nonfunctional requirements or from the application domain. Others will have to be elicited from the client. It is, however, necessary to state them explicitly such that every important design decision can be made consistently following the same set of criteria.
For example, in the light of the nonfunctional requirements for MyTrip, we identify reliability and
fault tolerance to connectivity loss as design goals. We then identify security as a design goal, as numerous drivers will have access to the same trip planning server. We add modifiability as a design goal, as we want to provide the ability for drivers to select a trip planning service of their choice. The following box summarizes the design goals we identified.
In general, we can select design goals from a long list of highly desirable qualities. Tables 5-1-5-5 list a number of possible design criteria. These criteria are organized into five groups: performance, dependability, cost, maintenance, and end user criteria. Performance, dependability,
and end user criteria are usually specified in the requirements or inferred from the application domain. Cost and maintenance criteria are dictated by the customer and the supplier.
Performance criteria (Table 5-1) include the speed and space requirements imposed on the system. Should the system be responsive or should it accomplish a maximum number of tasks? Is memory space available for speed optimizations or should memory be used sparingly?

Software Engineering module | PDF page 116 | lines 16-29

The analysis model describes the system completely from the actors' point of view and serves as the basis of communication between the client and the developers. The analysis model, however, does not contain information about the internal structure of the system, its hardware configuration,
or, more generally, how the system should be realized. System design is the first step in this direction. System design results in the following products:
• List of design goals, describing the qualities of the system that developers should optimize
• Software architecture, describing the subsystem decomposition in terms of subsystem responsibilities, dependencies among subsystems, subsystem mapping to hardware, and major policy decisions such as control flow, access control, and data storage.
The design goals are derived from the nonfunctional requirements. Design goals guide the decisions to be made by the developers especially when trade-offs are needed. The subsystem decomposition constitutes the bulk of system design. Developers divide the system into manageable pieces to deal with complexity: Each subsystem is assigned to a team and realized independently. In order for this to be possible, though, developers need to address system-wide

Question sources

Computer Science model exam collection · Question 81

Question 161 | SS176

In software risk management, accepting risk happening but don not do anything is known as:

A. Risk transfer

B. Risk Retention

C. Risk Reduction

D. Risk avoidance

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer B: Risk retention, specifically passive acceptance.

SE31 distinguishes removing a risk, reducing it, transferring consequences and retaining or accepting it. It explicitly says passive acceptance may take no action. The stem describes recognizing the risk and allowing it to remain without action, so retention matches. The other responses all involve a deliberate change to the risk or its consequences.

Why each choice is right or wrong

A - Incorrect: Transfer assigns consequences contractually rather than simply keeping the risk without action.

B - Correct: Retention or passive acceptance keeps the risk and may involve no immediate action.

C - Incorrect: Reduction actively lowers likelihood or impact instead of merely accepting exposure.

D - Incorrect: Avoidance changes the plan to remove the risk rather than allowing it to remain.

Primary note proof

Study-note section SE31

> Avoidance changes the plan to remove a risk, mitigation reduces likelihood/impact, transfer assigns consequences contractually, and retention/acceptance keeps the risk; passive acceptance may take no action. Resource leveling adjusts dates to respect resource availability and can change project finish; smoothing uses available float within schedule constraints. Neither magically creates extra staff.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 96 | lines 12-21

4.7. Risk Management
• Risk management is concerned with identifying risks and drawing up plans to minimise their effect on a project.
• A risk is a probability that some adverse circumstance will occur
- Project risks affect schedule or resources;
- Product risks affect the quality or performance of the software being developed;
- Business risks affect the organisation developing or procuring the software.
Risk Management Process
- Risk identification- Identify project, product and business risks;
- Risk analysis -Assess the likelihood and consequences of these risks;

Software Engineering module | PDF page 97 | lines 3-4

- Risk planning- Draw up plans to avoid or minimise the effects of the risk;
- Risk monitoring- Monitor the risks throughout the project.

Question sources

Computer Science model exam collection · Question 83

Question 162 | SS182

Which software process model will you use if you want to deliver different functionalities (modules) of the software product that have different priority
at different time?

A. Waterfall model

B. Spiral model

C. Incremental model

D. Linear model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Incremental model

Incremental development delivers prioritized functionality in usable stages. Earlier increments provide value while later increments add further capability. This is a delivery organization, distinct from a strict single-pass waterfall sequence. Spiral cycles are organized around risks and may use increments, but risk-driven development is not the specific name of the stated delivery strategy. A generic linear sequence also does not describe prioritized usable subsets.

Why each choice is right or wrong

A - Incorrect: Classical waterfall organizes phases before the principal usable delivery.

B - Incorrect: Spiral is defined by risk-driven cycles, even when an increment is produced.

C - Correct: Incremental development names the successive delivery of usable functionality.

D - Incorrect: A linear model does not name prioritized delivery of separate usable subsets.

Primary note proof

Study-note section SE04

> Choose incremental delivery when prioritized partial capability must reach users early. Smaller changes can be easier to debug and permit feedback, though architecture and integration still require planning. Iteration need not rebuild the whole system every time, and increments need not be perfectly independent modules. Broad descriptions of early working software can fit prototypes, iterative or Agile processes; identify the purpose of that early version.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 22 | lines 8-12

> This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
> also provide a platform for evaluation by the user.

Software Engineering module | PDF page 22 | lines 24-29

> Advantages of Incremental Model:
> • Generates working software quickly and early during the software life cycle. This model is more flexible - less costly to change scope and requirements. It is easier to test and debug during a smaller iteration.
> • In this model customer can respond to each built. Lowers initial delivery cost.
> • Easier to manage risk because risky pieces are identified and handled during its iteration.

Question sources

Computer Science model exam collection · Question 89

Question 163 | SS184

Which of the following is a resource optimization techniques in which start and finish dates are adjusted based on resource constraints with the goal
of balancing the demand for resources with the available supply?

A. Resource smoothing

B. Responsibility assignment matrix

C. Resource leveling

D. Resource grouping

Hint

Write out the supplied quantities and dependency constraints. Compute and compare values under the stated metric, finish condition and assumptions.

Answer C: Resource leveling.

SE31 defines resource leveling as adjusting dates to respect resource availability, potentially changing the project finish. That matches balancing resource demand against supply by moving starts and finishes. Smoothing instead works within available float and schedule constraints. A responsibility matrix assigns ownership, and resource grouping is not the defined date-adjustment technique.

Why each choice is right or wrong

A - Incorrect: Smoothing uses available float while retaining schedule constraints rather than generally leveling against availability.

B - Incorrect: A responsibility assignment matrix maps work to owners rather than adjusting task dates for resource capacity.

C - Correct: Resource leveling changes dates to balance required resources against available supply.

D - Incorrect: Resource grouping is not the defined schedule-adjustment method described by the stem.

Primary note proof

Study-note section SE31

> Avoidance changes the plan to remove a risk, mitigation reduces likelihood/impact, transfer assigns consequences contractually, and retention/acceptance keeps the risk; passive acceptance may take no action. Resource leveling adjusts dates to respect resource availability and can change project finish; smoothing uses available float within schedule constraints. Neither magically creates extra staff.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 95 | lines 3-12

4.6. Scheduling, Organization and Team Structures
Project-task scheduling is an important project planning activity. It involves deciding which tasks would
be taken up when. In order to schedule the project activities, a software project manager needs to do
the following:
• Identify all the tasks needed to complete the project and Break down large tasks into small activities.
• Determine the dependency among different activities and Establish the most likely estimates for the time durations necessary to complete the activities and Allocate resources to activities.
• Plan the starting and ending dates for various activities and Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

Question sources

Computer Science model exam collection · Question 91

Question 164 | SS190

Identify the design principle that does not apply to software systems

A. design should be structured to accommodate change

B. design should exhibit uniformity and integration

C. design should be reinventing the wheel from scratch

D. design should be traceable to the analysis model

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: design should be reinventing the wheel from scratch

Reusable classes, controlled change and traceability to requirements support software engineering. Recreating existing solutions from scratch contradicts its reuse rationale.

Why each choice is right or wrong

A - Incorrect: Low coupling explicitly limits future change impact. See PDF p. 32, 120, 70.

B - Incorrect: Consistent interfaces and integrated structures support the system, though this exact slogan is not quoted. See PDF p. 32, 120, 70.

C - Correct: Contradicts the documented promotion of reuse and component selection. See PDF p. 32, 120, 70.

D - Incorrect: Traceability explicitly supports validating design against requirements. See PDF p. 32, 120, 70.

Primary note proof

Study-note section SE25

Use abstraction and decomposition to manage complexity; preserve consistency, traceability and clear responsibilities. Top-down design refines high-level structure; bottom-up design assembles existing components. A hybrid combines both. SOLID includes Single Responsibility, Open-Closed, Liskov substitution, Interface Segregation and Dependency Inversion. Depend on abstractions; add compatible extensions; avoid forcing unused operations. DRY limits duplicated knowledge, KISS avoids unnecessary complexity, and YAGNI avoids speculative work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 32 | lines 12-21

A class in an object-oriented system carefully delineates between its interfaces the routines and attributes within a class are held together tightly. In a properly designed system, the classes will be grouped into subsystems but remain independent; therefore, changing one class has no impact
on other classes, and so, the impact is minimized. However, the object-oriented approach is not a panacea; nothing is magical here that will promote perfect design or perfect code.
Promotion of reusability
Objects are reusable because they are modeled directly out of a real-world problem domain. Each
object stands by itself or within a small circle of peers (other objects). Within this framework, the class does not concern itself with the rest of the system or how it is going to be used within a particular system.

Software Engineering module | PDF page 120 | lines 6-12

6.1.3. Coupling and coherence
Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Software Engineering module | PDF page 70 | lines 3-16

3.2.5. Verifiability and Traceability
Two more desirable properties of a system specification are that it be verifiable and traceable. The
specification is verifiable if, once the system is built; a repeatable test can be designed to demonstrate that the system fulfills the requirement. For example, a mean time to failure of a hundred years for SatWatch would be difficult to achieve (assuming it is realistic in the first place).
The following requirements are additional examples of non-verifiable requirements:
• The product shall have a good user interface (good is not defined).
• The product shall be error free (requires large amount of resources to establish).
• The product shall respond to the user with 1 second for most cases (“most cases” is not defined).

A system specification is traceable if each system function can be traced to its corresponding set of requirements. Traceability is not a constraint on the content of the specification, but rather, on its organization. Traceability facilitates the development of tests and the systematic validation of the design against the requirements.

Question sources

Computer Science model exam collection · Question 97

Question 165 | AM0001

Which phase of the software development life cycle is responsible for gathering and analyzing user requirements?

A. Analysis

B. Design

C. Development

D. Testing

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: Analysis

Requirements elicitation and analysis establish the system's purpose with users and formalize their needs; design and implementation follow.

Why each choice is right or wrong

A - Correct: The offered phase covering gathering and analysis. See PDF p. 62, 100.

B - Incorrect: Derives internal structure from requirements. See PDF p. 62, 100.

C - Incorrect: Produces executable code. See PDF p. 62, 100.

D - Incorrect: Detects faults in implemented behavior. See PDF p. 62, 100.

Primary note proof

Study-note section SE08

Business requirements explain why the project exists and its goals. User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints. The Software Requirements Specification (SRS) organizes the agreed requirements. A natural-language specification communicates with clients while analysis models formalize it for technical discussion; real specifications can also contain diagrams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 62 | lines 9-16

Requirement’s elicitation focuses on describing the purpose of the system. The client, the developers, and the users identify a problem area and define a system that addresses the problem.
Such a definition is called a system specification and serves as a contract between the client and the developers. The system specification is structured and formalized during analysis to produce an analysis model (see Figure 3.1). Both system specification and analysis model represent the same information. They differ only in the language and notation they use. The system specification
is written in natural language, whereas the analysis model is usually expressed in a formal or semiformal notation.

Software Engineering module | PDF page 100 | lines 11-24

Analysis focuses on producing a model of the system, called the analysis model, which is correct, complete, consistent, and verifiable. Analysis is different from requirements elicitation in that developers focus on structuring and formalizing the requirements elicited from users
Figure 4.1 Products of requirements elicitation and analysis (UML activity diagram).
Althoughtheanalysismodelmaynotbeunderstandabletotheusersandtheclient, it helps developers verify the system specification produced during requirements elicitation. There is a natural tendency for users and developers to postpone difficult decisions until later in the project. A decision may be difficult because of lack of domain knowledge, lack of technological knowledge, or simply because of disagreements among users and developers. Postponing decisions enables the
project to move on smoothly and avoid confrontation with reality or peers. Unfortunately, difficult
decisions will eventually need to be made, often at higher cost when intrinsic problems are discovered during testing, or worse, during user evaluation. Translating a system specification in to a formal or semi-formal model forces developers to identify and resolve difficult issues early in
the development.

Question sources

Computer Science MCQ practice collection · Question 1

Question 166 | AM0002

Which design principle suggests that software components should have a single, well-defined responsibility?

A. DRY (Don't Repeat Yourself) principle

B. SOLID principle

C. KISS (Keep It Simple, Stupid) principle

D. YAGNI (You Ain't Gonna Need It) principle

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: Single Responsibility, within SOLID.

SE25 includes Single Responsibility among the SOLID principles. That principle assigns a component one well-defined responsibility. DRY limits duplicated knowledge, KISS avoids unnecessary complexity and YAGNI avoids speculative work. Those can complement focused responsibilities, but none is the named principle directly described by the question.

Why each choice is right or wrong

A - Incorrect: DRY limits duplication of knowledge rather than specifically assigning one responsibility per component.

B - Correct: SOLID includes the Single Responsibility principle that the stem describes.

C - Incorrect: KISS favors simplicity, a different design concern from the specific single-responsibility rule.

D - Incorrect: YAGNI avoids speculative work rather than defining responsibility boundaries.

Primary note proof

Study-note section SE25

> Use abstraction and decomposition to manage complexity; preserve consistency, traceability and clear responsibilities. Top-down design refines high-level structure; bottom-up design assembles existing components. A hybrid combines both. SOLID includes Single Responsibility, Open-Closed, Liskov substitution, Interface Segregation and Dependency Inversion. Depend on abstractions; add compatible extensions; avoid forcing unused operations. DRY limits duplicated knowledge, KISS avoids unnecessary complexity, and YAGNI avoids speculative work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 120 | lines 24-27

> Coherence is the strength of dependencies within a subsystem. If a subsystem contains many objects that are related to each other and perform similar tasks, its coherence is high. If a subsystem
> contains a number of unrelated objects, its coherence is low. A desirable property of a subsystem decomposition is that it leads to subsystems with high coherence.

Question sources

Computer Science MCQ practice collection · Question 2

Question 167 | AM0003

Which of the following is an agile development methodology?

A. Waterfall model

B. Scrum

C. V-model

D. Spiral model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Scrum.

SE05 presents Scrum through timeboxed Sprints, a Sprint Goal and a usable Increment, in the Agile context. Waterfall and the V-model organize planned stages and corresponding tests, while spiral is defined by risk-driven cycles. Therefore Scrum is the specifically Agile approach among the listed alternatives; iteration alone does not make every process Scrum.

Why each choice is right or wrong

A - Incorrect: Classical waterfall organizes planned phases rather than the Agile Scrum approach described.

B - Correct: Scrum is the Agile framework using Sprints and usable Increments.

C - Incorrect: The V-model pairs development stages with test levels rather than being the listed Agile framework.

D - Incorrect: Spiral is risk driven and is not the specifically Agile methodology named among these choices.

Primary note proof

Study-note section SE05

> Scrum uses timeboxed Sprints of at most one month, with a Sprint Goal, selected work and a usable Increment meeting a Definition of Done. Scope is discussed before and during planning; it is not forbidden to agree scope. Timeboxes tend to stay fixed while content is adapted without endangering the goal. Agile can be combined with required assurance/documentation; safety-critical context alone cannot prove it is unsuitable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

> Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules

to
follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Question sources

Computer Science MCQ practice collection · Question 3

Question 168 | AM0004

A team delivers small usable capabilities, replans the next work with customers and treats adaptation as a central value rather than organizing cycles around risk analysis. Which broad approach best fits?

A. Waterfall model

B. Agile methodology

C. Prototype model

D. Spiral model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Agile methodology

The team adds small usable capabilities and repeatedly replans with customers. Agile makes this combination of working results, customer participation and adaptation central. A prototype's immediate purpose is learning about requirements or feasibility; a spiral cycle is organized around risk reduction. Classical waterfall instead emphasizes planned phase commitments. Increments by themselves can occur in several processes, so the collaboration and planning priorities establish the intended comparison.

Why each choice is right or wrong

A - Incorrect: Classical waterfall does not make repeated adaptive replanning its organizing emphasis.

B - Correct: Agile combines working increments with ongoing customer collaboration and adaptation.

C - Incorrect: Prototyping primarily explores uncertainty rather than naming this ongoing delivery-and-planning organization.

D - Incorrect: Spiral's defining emphasis is risk analysis and reduction.

Primary note proof

Study-note section SE04

Iteration revisits and improves a solution; an increment adds usable functionality. A process may do both. Delivering registration first, then payments, then reporting is incremental; revising

a registration prototype after feedback is iterative. Each increment still needs design, implementation and testing.

Study-note section SE06

Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Study-note section SE05

Agile favors people collaborating, useful working results, customer participation and adaptation over rigid adherence to procedures, exhaustive documents, contracts or fixed plans. The latter still have value. Agile does not mean no planning or no documentation. A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to
follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Software Engineering module | PDF page 22 | lines 8-12

This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
also provide a platform for evaluation by the user.

Question sources

Computer Science MCQ practice collection · Question 4

Question 169 | AM0005

Which software testing technique focuses on finding defects by executing the software without knowledge of its internal structure?

A. White-box testing

B. Black-box testing

C. Integration testing

D. System testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: Black-box testing

Black-box equivalence testing chooses input classes based on expected behavior without tracing internal paths; white-box paths require source knowledge.

Why each choice is right or wrong

A - Incorrect: Requires code/data-structure knowledge for path selection. See PDF p. 149, 149.

B - Correct: Matches behavior testing without internal-structure knowledge. See PDF p. 149, 149.

C - Incorrect: A test level that can combine techniques. See PDF p. 149, 149.

D - Incorrect: A complete-system scope rather than the knowledge-based distinction. See PDF p. 149, 149.

Primary note proof

Study-note section SE36

> Black-box techniques derive cases from specified behavior; white-box techniques derive them from internal structure; grey-box uses partial internal knowledge. Regression testing checks for unintended effects after change; confirmation testing checks the particular defect repair. Exploratory testing combines learning, design and execution and is experience based. Manual/automated describe execution support, not a compulsory black-/white-box classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 149 | lines 6-10

> Equivalence testing is a black box testing technique that minimizes the number of test cases. The possible inputs are partitioned into equivalence classes, and a test case is selected for each class. The assumption of equivalence testing is that systems usually behave in similar ways for all members of a class. To test the behavior associated with an equivalence class, we only need to test
> one member of the class. Equivalence testing consists of two steps: identification of the

Software Engineering module | PDF page 149 | lines 24-29

Path testing
Path testing is a white box testing technique that identifies faults in the implementation of the component. The assumption behind path testing is that, by exercising all possible paths through the code at least once, most faults will trigger failures. The identification of paths requires knowledge of the source code and data structures.
The starting point for path testing is the flow graph. A flow graph consists of nodes representing

Question sources

Computer Science MCQ practice collection · Question 5

Question 170 | AM0006

Which diagrams describe detailed behavioral evolution over time through either ordered participant interactions or event-triggered state changes? Select all that apply.

A. Use case diagrams

B. Class diagrams

C. Sequence diagrams

D. State machine diagrams

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C, D: Sequence diagrams; State machine diagrams

Sequence diagrams show ordered messages across participant lifelines, while state machines show event-triggered transitions through an object's lifecycle states. Both therefore describe behavioral evolution over time under the stated scope. A use-case diagram summarizes external roles and goals without that detailed temporal evolution, and a class diagram describes static types and relationships. Neither replaces the two detailed dynamic views requested.

Why each choice is right or wrong

A - Incorrect: Use-case diagrams summarize roles/goals rather than detailed temporal evolution.

B - Incorrect: Class diagrams are static.

C - Correct: Sequence diagrams show ordered participant interactions.

D - Correct: State machines show event-triggered lifecycle changes.

Primary note proof

Study-note section SE20

Choose a sequence diagram for message order between participants, a state machine for event-dependent state changes, and activity for workflow. Both sequence and state diagrams describe behavior over time. State machines are useful for objects with significant states,

whether entity or control. A DFD shows data stores/flows and transformations from inputs to outputs; it is not a UML diagram.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 54 | lines 26-35

both buttons simultaneously. Figure 2.10 depicts a sequence diagram for an actor setting his 2Bwatch one minute ahead.
Each column represents an object that participates in the interaction. The vertical axis represents
time from top to bottom. Messages are shown by arrows. Labels on arrows represent message names and may contain arguments. Activations (i.e., executing methods) are depicted by vertical rectangles. Actors are shown as the leftmost column.
Sequence diagrams can be used to describe either an abstract sequence (i.e., all possible interactions) or concrete sequences (i.e., one possible interaction, as in Figure 2.10). When describing all possible interactions, sequence diagrams also provide notations for conditionals and
iterators.

Software Engineering module | PDF page 55 | lines 4-9

2.3.3. State chart Diagram
A UML statechart is a notation for describing the sequence of states an object goes through in response to external events. Statecharts are extensions of the finite state machine model. On the one hand, statecharts provide notation for nesting states and state machines (i.e., a state can be described by a state machine). On the other hand, statecharts provide notation for binding transitions with message sends and conditions on objects.

Question sources

Computer Science MCQ practice collection · Question 6

Question 171 | AM0007

Which software development methodology promotes adaptive planning, evolutionary development, and continuous improvement?

A. Waterfall model

B. Agile methodology

C. V-model

D. Spiral model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Agile methodology

Adaptive planning, evolutionary improvement and responsiveness to customer feedback characterize Agile as a broad development approach. Waterfall emphasizes planned phase progression; the V-model pairs development stages with corresponding test levels. Spiral can evolve a product as well, but its defining organization is repeated risk assessment and reduction. The question asks for the approach whose core emphasis matches the stated planning and collaboration values.

Why each choice is right or wrong

A - Incorrect: Waterfall emphasizes planned phase progression rather than adaptive planning.

B - Correct: Agile places adaptive planning and ongoing improvement at the center.

C - Incorrect: The V-model emphasizes correspondence between development and test stages.

D - Incorrect: Spiral is organized primarily around risk reduction, even when development is evolutionary.

Primary note proof

Study-note section SE05

> Agile favors people collaborating, useful working results, customer participation and adaptation over rigid adherence to procedures, exhaustive documents, contracts or fixed plans. The latter still have value. Agile does not mean no planning or no documentation. A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

> Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to

follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Software Engineering module | PDF page 22 | lines 8-12

This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
also provide a platform for evaluation by the user.

Question sources

Computer Science MCQ practice collection · Question 7

Question 172 | AM0008

Which technique is used to represent the flow of activities and decisions in a software system?

A. Use case modeling

B. Activity diagrams

C. State machine diagrams

D. Component diagrams

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: Activity diagrams

Activity diagrams interpret actions and transitions as control flow; decision diamonds branch that flow based on conditions.

Why each choice is right or wrong

A - Incorrect: Captures external functions, not detailed action flow. See PDF p. 56, 57.

B - Correct: Explicit action/control-flow notation with decisions. See PDF p. 56, 57.

C - Incorrect: Shows states responding to events rather than primarily the activity sequence. See PDF p. 56, 57.

D - Incorrect: Represents component organization rather than activity decisions. See PDF p. 56, 57.

Primary note proof

Study-note section SE20

An activity diagram models workflow: filled circle start, rounded actions, guarded decision diamond, thick fork/join bars and partitions/swimlanes for responsibility. A diamond can also

merge alternative flows; a join synchronizes concurrent flows. A state machine models states and transitions triggered by events, often with guards/actions, for one object's lifecycle.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 56 | lines 18-24

2.3.4. Activity diagrams
The outgoing transitions are triggered by the completion of an action associated with the state. This is called an action state. By convention, the name of a state denotes a condition, whereas the name of an action state denotes an action. Activity diagrams are statechart diagrams whose states
are action states. is an activity diagram corresponding to the state diagram. An alternate and equivalent view of activity diagrams is to interpret action states as control flow between activities
and transitions; that is, the arrows are interpreted as sequential constraints between activities.

Software Engineering module | PDF page 57 | lines 7-11

Decisions are branches in the control flow. They denote alternative transitions based on a condition
of the state of an object or a set of objects. Decisions are depicted by a diamond with one or more incoming arrows and two or more outgoing arrows. The outgoing arrows are labeled with the conditions that select a branch in the control flow. The set of all outgoing transitions from a decision represents the set of all possible outcomes.

Question sources

Computer Science MCQ practice collection · Question 8

Question 173 | AM0009

Developers build a quick interface version solely to discover how users perform an unfamiliar task; it is not a production release. Which named approach describes the immediate purpose of that version?

A. Waterfall model

B. Agile methodology

C. Prototype model

D. Spiral model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Prototype model

The early version is built to learn how users understand and carry out a task, not to deliver a production increment. That exploratory purpose identifies a prototype. Agile can use prototypes but its broader definition centers on collaboration and adaptation. Spiral may prototype to reduce a risk, but risk cycles are not the specific technique requested. Classical waterfall does not itself name early experimental versions. The purpose of the artifact distinguishes these overlapping practices.

Why each choice is right or wrong

A - Incorrect: Waterfall is a phased lifecycle rather than the named exploratory artifact technique.

B - Incorrect: Agile may use prototypes but does not specifically name the described usability experiment.

C - Correct: Prototyping uses an experimental version to discover requirements or usability needs.

D - Incorrect: Spiral may employ a prototype, but its defining method is organizing cycles around risk reduction.

Primary note proof

Study-note section SE07

> A prototype explores requirements, usability or technical feasibility. A throwaway prototype is discarded after learning; an evolutionary prototype is deliberately refined toward production. A quick prototype may omit robustness, security or architecture, so do not deploy it unchanged without evaluation. Coding may occur during prototyping even if a diagram labels that activity construction rather than coding.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 82 | lines 23-29

> • Prototype test. During this type of test, one or more users are presented with a piece of software that practically implements the system. A vertical prototype implements a complete use case through the system, and a horizontal prototype presents an interface for most use cases (without providing much or any functionality). The advantages of prototype tests are that they provide a realistic view of the system to the user and that prototypes can be instrumented to collect detailed data. The disadvantages of prototypes are that they are expensive to build and to modify.

Software Engineering module | PDF page 26 | lines 6-9

2. Requirement’s elicitation and analysis: This is the process of deriving the system requirements through observation of existing systems, discussions with potential users and procurers, task analysis, and so on. This may involve the development of one or more system models and prototypes. These help you understand the system to be specified.

Question sources

Computer Science MCQ practice collection · Question 9

Question 174 | AM0010

Which testing technique focuses on testing the interaction between different components or modules of a software system?

A. Unit testing

B. Integration testing

C. System testing

D. Regression testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: Integration testing

Integration testing targets interfaces and interactions among components that have been combined. Unit testing isolates a smaller function or component, while system testing evaluates the complete integrated product against its requirements. Regression testing is triggered by a change and looks for unintended effects; it can be performed at several levels. The scenario is categorized by its interaction scope, so integration is the specific level requested.

Why each choice is right or wrong

A - Incorrect: Unit testing isolates a function or small component.

B - Correct: Integration testing exercises interactions among combined components.

C - Incorrect: System testing evaluates the complete product rather than specifically the component interfaces.

D - Incorrect: Regression names the after-change purpose, not a particular component-combination level.

Primary note proof

Study-note section SE35

Unit testing isolates a function, class or small component, often replacing dependencies with stubs/mocks; a driver invokes the unit and a stub substitutes a called dependency. Integration testing exercises interfaces and interactions among combined parts. System testing checks the

complete integrated system against functional and nonfunctional requirements in a representative environment.

Study-note section SE36

Functional tests check required behavior against the SRS and acceptance criteria. Performance tests assess speed, throughput and resources under a specified workload; load uses expected demands and stress exceeds normal limits. Usability, security, compatibility and reliability tests target their respective qualities. “Normal operating conditions” alone does not distinguish functional from performance testing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Software Engineering module | PDF page 147 | lines 14-18

• Unit testing, which finds faults by isolating an individual component using test stubs and drivers and by exercising the component using a test case
• Integration testing, which finds faults by integrating several components together
• System testing, which focuses on the complete system, its functional and nonfunctional requirements, and its target environment

Question sources

Computer Science MCQ practice collection · Question 10

Question 175 | AM0011

Which design principle suggests that software entities should depend on abstractions, not on concrete implementations?

A. DRY (Don't Repeat Yourself) principle

B. SOLID principle

C. KISS (Keep It Simple, Stupid) principle

D. YAGNI (You Ain't Gonna Need It) principle

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: Dependency Inversion, within SOLID.

SE25 lists Dependency Inversion within SOLID and explicitly directs designs to depend on abstractions. That reduces dependence on concrete implementation details and matches the stem. DRY, KISS and YAGNI address duplication, simplicity and speculative functionality respectively; none is the principle defining the abstraction-based dependency relationship.

Why each choice is right or wrong

A - Incorrect: DRY addresses duplicated knowledge rather than the direction and target of dependencies.

B - Correct: SOLID includes Dependency Inversion, which directs dependencies toward abstractions.

C - Incorrect: KISS addresses unnecessary complexity rather than specifying abstraction-based dependencies.

D - Incorrect: YAGNI limits speculative work rather than defining dependency direction.

Primary note proof

Study-note section SE25

> Use abstraction and decomposition to manage complexity; preserve consistency, traceability and clear responsibilities. Top-down design refines high-level structure; bottom-up design assembles existing components. A hybrid combines both. SOLID includes Single Responsibility, Open-Closed, Liskov substitution, Interface Segregation and Dependency Inversion. Depend on abstractions; add compatible extensions; avoid forcing unused operations. DRY limits duplicated knowledge, KISS avoids unnecessary complexity, and YAGNI avoids speculative work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 119 | lines 13-25

> 6.1.2. Services and subsystem interfaces
> A subsystem is characterized by the services it provides to other subsystems. A service is a set of

related operations that share a common purpose. A subsystem providing a notification service, for
example, defines operations to send notices, look up notification channels, and subscribe and unsubscribe to a channel.
The set of operations of a subsystem that are available to other subsystems form the subsystem interface. The subsystem interface, also referred to as the application programmer interface (API),
includes the name of the operations, their parameters, their types, and their return values. System
design focuses on defining the services provided by each subsystem that is, enumerating the operations, their parameters, and their high-level behavior. Object design will focus on defining the subsystem interfaces, that is, the type of the parameters and the return value of each operation.
The definition of a subsystem in terms of the services it provides helps us focus on its interface as opposed to its implementation. A good subsystem interface should provide as little information

Software Engineering module | PDF page 120 | lines 6-12

6.1.3. Coupling and coherence
Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Question sources

Computer Science MCQ practice collection · Question 11

Question 176 | AM0012

Which software development approach follows a linear, sequential flow from requirements gathering to testing and maintenance?

A. Waterfall model

B. Agile methodology

C. Prototype model

D. Spiral model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A: Waterfall model

Classical waterfall organizes development as planned sequential phases such as requirements, design, implementation and testing. Agile instead adapts work through feedback, prototyping uses experimental versions to learn, and spiral cycles organize work around risk reduction. Real waterfall projects can revisit work under change control, but the idealized linear organization described by the question is its characteristic model rather than one of the feedback-centered alternatives.

Why each choice is right or wrong

A - Correct: The idealized waterfall model follows the stated planned phase sequence.

B - Incorrect: Agile emphasizes adaptation and incremental planning.

C - Incorrect: Prototyping learns through experimental versions and feedback.

D - Incorrect: Spiral organizes repeated cycles around risk assessment and reduction.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 15 | lines 16-20

> non-iterative (or sequential) process in which progress is seen as flowing steadily downwards (like
> a waterfall) through the development phases. Thus the waterfall model maintains that one should
> move to a phase only when its preceding phase is reviewed and verified. That is, each phase must
> be completed before the next phase can begin. In waterfall model phases do not overlap. But there
> are various modified waterfall models, and hence, may include slight or major modifications.

Question sources

Computer Science MCQ practice collection · Question 12

Question 177 | AM0013

Which technique is used to estimate the size and complexity of a software project?

A. Use case modeling

B. Storyboarding

C. Function point analysis

D. Pair programming

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer C: Function point analysis

Function-point analysis counts system externals, assigns complexity weights and aggregates them to estimate software size independent of implementation language.

Why each choice is right or wrong

A - Incorrect: Captures functionality but is not the stated weighted size-estimation method. See PDF p. 93, 91.

B - Incorrect: A user-experience representation, not the module's quantitative size method. See PDF p. 93, 91.

C - Correct: Directly taught size/complexity weighting technique. See PDF p. 93, 91.

D - Incorrect: A development collaboration practice, not this sizing method. See PDF p. 93, 91.

Primary note proof

Study-note section SE28

> Function points estimate functional size using inputs, outputs, inquiries, internal logical files and external interface files, weighted under the chosen method. They are not just lines of code and are not a direct exact measure of complexity. Git tracks versions; issue trackers such as Jira track work/defects; CI tools such as Jenkins automate checks; Docker packages application/runtime dependencies into containers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 93 | lines 3-21

> • Function points allow the measurement of software size in standard units, independent of the underlying language in which the software is developed.
> • Counting the number of externals (inputs, outputs, inquiries, and interfaces) that make up the system.
> • There are five types of externals to count:
> 1. External inputs - data or control inputs (input files, tables, forms, screens, messages,

etc.) to the system
2. External outputs - data or control outputs from the system
3. External inquiries - I/O queries which require a response (prompts, interrupts, calls, etc.)
4. External interfaces - libraries or programs which are passed into and out of the system (I/O routines, sorting procedures, math libraries, run-time libraries, etc.)
5. Internal data files - groupings of data stored internally in the system (entities, internal control files, directories)
Apply these steps to calculate the size of a project:
1. Count or estimate all the occurrences of each type of external.
2. Assign each occurrence a complexity weight.
3. Multiply each occurrence by its complexity weight, and total the results to obtain a function count. Complexity weights are as listed below.

Software Engineering module | PDF page 91 | lines 37-42

• Software Metrics can be used to indicate the basic attributes of the software. The indicative metrics, not exhaustive, are
• Size - Line of code ,Function point count
• Quality- Reliability, Accuracy, Dependability, and these are measured through some measures like errors, failures and number of runs between failures ,etc.
• Execution Time -process time of execution of a critical process and many more.

Question sources

Computer Science MCQ practice collection · Question 13

Question 178 | AM0014

Which tool is used to track and manage software issues, bugs, and tasks throughout the development process?

A. Git

B. Jira

C. Jenkins

D. Docker

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer B: Jira.

SE28 distinguishes version tracking, issue tracking, automated checks and container packaging. It names Jira as an issue tracker for work and defects, matching bugs and tasks across development. Git records versions, Jenkins automates checks and Docker packages runtime dependencies; those tools may integrate with issue tracking but have different primary roles.

Why each choice is right or wrong

A - Incorrect: Git tracks source versions and history rather than serving as the named issue/task tracker.

B - Correct: Jira tracks work items, defects and tasks through their development workflow.

C - Incorrect: Jenkins automates checks and integration work rather than primarily managing issue records.

D - Incorrect: Docker packages runtime dependencies into containers rather than tracking bugs and tasks.

Primary note proof

Study-note section SE28

> Function points estimate functional size using inputs, outputs, inquiries, internal logical files and external interface files, weighted under the chosen method. They are not just lines of code and are not a direct exact measure of complexity. Git tracks versions; issue trackers such as Jira track work/defects; CI tools such as Jenkins automate checks; Docker packages application/runtime dependencies into containers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 142 | lines 5-10

> Configuration management avoids faults caused by undisciplined change in the system models. For example, it is a common mistake to a change a subsystem interface without notifying all developers of calling components. Configuration management can make sure that, if analysis models and code are becoming inconsistent with one another, analysts and implementor are notified. The assumption behind configuration management is that the system contains many less
> faults if change is controlled.

Software Engineering module | PDF page 90 | lines 7-24

> 4.2. Project Planning
> • Probably the most time-consuming project management activity.
> • Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
> • Various different types of plans may be developed to support the main software project plan that is concerned with schedule and budget.
> The project plan sets out:
> • The resources available to the project;
> • The work breakdown;
> • A schedule for the work.
> Project Plan Structure
> • Introduction.
> • Project organisation.
> • Risk analysis.
> • Hardware and software resource requirements.

• Work breakdown.
• Project schedule.
• Monitoring and reporting mechanisms.

Question sources

Computer Science MCQ practice collection · Question 14

Question 179 | AM0015

Which design principle suggests that software entities should be open for extension but closed for modification?

A. DRY (Don't Repeat Yourself) principle

B. SOLID principle

C. KISS (Keep It Simple, Stupid) principle

D. YAGNI (You Ain't Gonna Need It) principle

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: Open-Closed, within SOLID.

SE25 lists Open-Closed within SOLID and recommends adding compatible extensions. The stem quotes that principle's central idea: support extension while keeping established code stable. DRY, KISS and YAGNI address other concerns. Open-Closed is a design goal achieved through suitable abstractions, rather than a claim that software can never need modification.

Why each choice is right or wrong

A - Incorrect: DRY limits duplicated knowledge rather than defining openness for extension.

B - Correct: SOLID includes the Open-Closed principle explicitly described by the stem.

C - Incorrect: KISS favors simple solutions rather than the specific extension-versus-modification relationship.

D - Incorrect: YAGNI avoids speculative functionality rather than defining the Open-Closed design goal.

Primary note proof

Study-note section SE25

Use abstraction and decomposition to manage complexity; preserve consistency, traceability and clear responsibilities. Top-down design refines high-level structure; bottom-up design assembles existing components. A hybrid combines both. SOLID includes Single Responsibility, Open-Closed, Liskov substitution, Interface Segregation and Dependency Inversion. Depend on

abstractions; add compatible extensions; avoid forcing unused operations. DRY limits duplicated knowledge, KISS avoids unnecessary complexity, and YAGNI avoids speculative work.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 104 | lines 6-17

5.1.4. Generalization
Generalization enables us to organize concepts in to hierarchies. There may be any number of intermediate levels in between, covering more-or-less generalized concepts (e.g., Low Priority Incident, Emergency, Disaster). Such hierarchies allow us to refer to many concepts precisely. When
we use the term Incident, we mean all instances of all types of Incidents. When we use the term Emergency, we only refer to an Incident that requires an immediate response. This view of generalization stems from modeling.
In an object-oriented programming language, inheritance is a reusability technique. If a class Child
inherits from a class Parent, all the attributes and methods available on Parent are automatically available on Child. The class Child may add additional methods or override inherited methods, thus refining the class Parent. As in the case of generalization, the class at the top of the hierarchy
tends to be the most general one, whereas the leaves tend to be the most specialized.

Software Engineering module | PDF page 32 | lines 12-21

A class in an object-oriented system carefully delineates between its interfaces the routines and attributes within a class are held together tightly. In a properly designed system, the classes will be grouped into subsystems but remain independent; therefore, changing one class has no impact
on other classes, and so, the impact is minimized. However, the object-oriented approach is not a panacea; nothing is magical here that will promote perfect design or perfect code.
Promotion of reusability
Objects are reusable because they are modeled directly out of a real-world problem domain. Each
object stands by itself or within a small circle of peers (other objects). Within this framework, the class does not concern itself with the rest of the system or how it is going to be used within a particular system.

Question sources

Computer Science MCQ practice collection · Question 15

Question 180 | AM0016

Which technique is used to represent the static structure of a software system, including classes, relationships, and attributes?

A. Use case diagrams

B. Class diagrams

C. Sequence diagrams

D. State machine diagrams

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: Class diagrams

Class diagrams explicitly depict classes with attributes and operations and their associations/generalizations.

Why each choice is right or wrong

A - Incorrect: External functions and actors. See PDF p. 51.

B - Correct: Exact structural modeling notation. See PDF p. 51.

C - Incorrect: Time-ordered messages among objects. See PDF p. 51.

D - Incorrect: State evolution rather than static attributes/relationships. See PDF p. 51.

Primary note proof

Study-note section SE17

> A class box lists name, attributes and operations; + public, - private, # protected and ~ package indicate visibility. A class diagram shows static types and relationships, while an object diagram shows particular instances/links. UML is maintained and standardized by OMG; it is a modeling language, not a lifecycle process. Structural things describe conceptual or physical structural elements.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 51 | lines 14-24

> 2.3.2. Class Diagrams
> Classes and objects
> Class diagrams describe the structure of the system in terms of classes and objects. Classes are abstractions that specify the attributes and behavior of a set of objects. Objects are entities that encapsulate state and behavior. Each object has an identity: It can be referred individually and is distinguishable from other objects.
> In UML, classes and objects are depicted by boxes including three compartments. The top compartment displays the name of the class or object. The center compartment displays its

attributes, and the bottom compartment displays its operations. The attribute and operation compartment can be omitted for clarity. Object names are underlined to indicate that they are instances. By convention, class names start with an uppercase letter. Objects in object diagrams

Question sources

Computer Science MCQ practice collection · Question 16

Question 181 | AM0017

Customer representatives stay in close contact with developers and use feedback to reshape the next timeboxed work. Which methodology makes this collaboration and adaptation a foundational priority?

A. Waterfall model

B. Agile methodology

C. V-model

D. Spiral model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Agile methodology

Frequent stakeholder communication alone is possible in several lifecycles. The scenario additionally gives customer representatives ongoing access to the team and lets feedback reshape the next timeboxed body of work, matching Agile's collaboration and adaptation values. Waterfall and the V-model primarily organize planned stages, while spiral organizes commitments around risk reduction. The combination of priorities, rather than communication frequency by itself, identifies the intended methodology.

Why each choice is right or wrong

A - Incorrect: Waterfall is defined by planned phase progression, not the stated adaptive collaboration priority.

B - Correct: Agile explicitly prioritizes customer collaboration and responding to change.

C - Incorrect: The V-model is defined by linked development and testing stages.

D - Incorrect: Spiral's organizing priority is risk assessment and reduction, though stakeholders can participate.

Primary note proof

Study-note section SE05

Agile favors people collaborating, useful working results, customer participation and adaptation over rigid adherence to procedures, exhaustive documents, contracts or fixed plans. The latter

still have value. Agile does not mean no planning or no documentation. A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Study-note section SE03

Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Study-note section SE06

Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to
follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Software Engineering module | PDF page 22 | lines 8-12

This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
also provide a platform for evaluation by the user.

Software Engineering module | PDF page 81 | lines 11-21

3.4.2. Negotiating specifications with clients: Joint Application Design
Joint Application Design (JAD) is effectiveness lies in that the requirements elicitation work is done in one single workshop session in which all stakeholders participate. Users, clients, developers, and a trained session leader sit together in one room to present their viewpoint, listen
to other viewpoints, negotiate, and agree on a mutually acceptable solution. The outcome of the workshop, the final JAD document, is a complete system specification document that includes definitions of data elements, work flows, and interface screens. Because the final document is jointly developed by the stakeholders (that is, the participants who not only have an interest in the

success of the project, but also can make substantial decisions) the final JAD document represents
an agreement between users, clients, and developers, and thus minimizes requirements changes later in the development process.

Question sources

Computer Science MCQ practice collection · Question 17

Question 182 | AM0018

Which technique documents an actor’s goal with preconditions, a main success flow, alternative flows and postconditions?

A. Use case modeling

B. Storyboarding

C. Wireframing

D. Pair programming

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer A: Use case modeling

A use-case description captures an actor’s goal together with preconditions, main flow, alternatives and postconditions. This provides the structured behavioral requirements record in the scenario. Storyboards illustrate moments in an interaction and wireframes sketch interface structure; either can help discovery without supplying that full behavioral contract. Pair programming organizes two people’s implementation work. It is therefore use-case modeling that directly names the requested requirements representation.

Why each choice is right or wrong

A - Correct: Use-case modeling records actors, goals and structured successful and alternative flows.

B - Incorrect: Storyboarding sketches interaction moments rather than necessarily supplying the full goal-and-flow description.

C - Incorrect: Wireframing sketches interface structure and layout.

D - Incorrect: Pair programming is collaborative implementation work.

Primary note proof

Study-note section SE14

An actor is an external role interacting with the system; it can be a person, device or another system. A person may play several roles, and several people may play one role. A primary actor

seeks a goal; a supporting actor provides a service. Ask which user groups perform main functions, administer or maintain the system, and which external systems exchange information.

Study-note section SE-S05

A storyboard sketches a sequence of moments in a proposed interaction, helping people explore how a concept would be used. A wireframe sketches interface structure and the arrangement of content or controls. Both can inform requirements, but neither automatically supplies a complete use-case description with actors, preconditions, flows and postconditions.

Study-note section SE-S05

Pair programming is a way for two programmers to work together on the same implementation task, commonly sharing a workstation with complementary writing and reviewing roles. It is an implementation collaboration practice rather than a replacement for modeling a user goal.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 47 | lines 30-39

Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
Use cases describe the behavior of the system as seen from an actor's point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate a
use case to access the system functionality. The use case can then initiate other use cases and gather
more information from the actors. When actors and use cases exchange information, they are said
to communicate. We will see later that we represent these exchanges with communication relationships.

Software Engineering module | PDF page 48 | lines 18-32

To describe a use case, we use a template composed of six fields (see also Figure 2.2):
• The name of the use case is unique across the system so that developers (and project participants) can unambiguously refer to the use case.
• Participating actors are actors interacting with the use case.
• Entry conditions describe the conditions that need to be satisfied before the use case is initiated.
• The flow of events describes the sequence of actions of the use case, which are be numbered for reference. The common case (i.e., cases that occur frequently) and the exceptional cases (i.e., cases that seldom occur, such as errors and unusual conditions) are described separately in different use cases for clarity.
• Exit conditions describe the conditions that are satisfied after the completion of the use case.
• Special requirements are requirements that are not related to the functionality of the

system. These include constraints on the performance of the system, its implementation, the hardware platforms it runs on, and so on.

Question sources

Computer Science MCQ practice collection · Question 18

Question 183 | AM0019

Which testing technique focuses on retesting modified or affected parts of the software to ensure that changes do not introduce new defects?

A. Unit testing

B. Integration testing

C. Regression testing

D. System testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer C: Regression testing.

SE36 defines regression testing by the purpose of detecting unintended effects after change. That includes retesting changed and affected areas to find newly introduced faults. Unit, integration and system testing describe levels or scope; regression tests can be run at any of them, so the change-driven purpose uniquely identifies C.

Why each choice is right or wrong

A - Incorrect: Unit testing defines a small-component scope rather than the purpose of checking change-induced defects.

B - Incorrect: Integration testing focuses on interfaces and interactions, which may be regression-tested but do not define regression.

C - Correct: Regression testing retests previously working behavior to detect unintended effects of modifications.

D - Incorrect: System testing addresses the complete integrated system rather than specifically the after-change purpose.

Primary note proof

Study-note section SE36

Black-box techniques derive cases from specified behavior; white-box techniques derive them from internal structure; grey-box uses partial internal knowledge. Regression testing checks for unintended effects after change; confirmation testing checks the particular defect repair.

Exploratory testing combines learning, design and execution and is experience based. Manual/automated describe execution support, not a compulsory black-/white-box classification.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 147 | lines 14-18

• Unit testing, which finds faults by isolating an individual component using test stubs and drivers and by exercising the component using a test case
• Integration testing, which finds faults by integrating several components together
• System testing, which focuses on the complete system, its functional and nonfunctional requirements, and its target environment

Software Engineering module | PDF page 142 | lines 5-10

Configuration management avoids faults caused by undisciplined change in the system models. For example, it is a common mistake to a change a subsystem interface without notifying all developers of calling components. Configuration management can make sure that, if analysis models and code are becoming inconsistent with one another, analysts and implementor are notified. The assumption behind configuration management is that the system contains many less
faults if change is controlled.

Software Engineering module | PDF page 147 | lines 3-8

• A test stub is a partial implementation of components on which the tested component depends. A test driver is a partial implementation of a component that depends on the tested component. Test stubs and drivers enable components to be isolated from the rest of the system for testing.
• A correction is a change to a component. The purpose of a correction is to repair a fault. Note that a correction can introduce new faults.

Question sources

Computer Science MCQ practice collection · Question 19

Question 184 | AM0020

Which tool is used to package an application and its dependencies into a container for easy deployment and scalability?

A. Git

B. Jira

C. Jenkins

D. Docker

Hint

Distinguish the planning or organization goal from the tool or structure used to support it. Apply the stated project conditions.

Answer D: Docker.

SE28 explicitly says Docker packages an application and its runtime dependencies into containers. That matches the deployment packaging task in the question. Git tracks versions, Jira tracks issues and Jenkins automates checks; they can participate in a deployment pipeline, but none replaces Docker's specified container packaging responsibility.

Why each choice is right or wrong

A - Incorrect: Git manages version history rather than packaging the runtime into a container.

B - Incorrect: Jira manages work and defects rather than building application containers.

C - Incorrect: Jenkins automates checks and pipelines but is not the named container-packaging tool.

D - Correct: Docker packages the application and runtime dependencies into a deployable container.

Primary note proof

Study-note section SE28

> Function points estimate functional size using inputs, outputs, inquiries, internal logical files and external interface files, weighted under the chosen method. They are not just lines of code and are not a direct exact measure of complexity. Git tracks versions; issue trackers such as Jira track work/defects; CI tools such as Jenkins automate checks; Docker packages application/runtime dependencies into containers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 124 | lines 8-16

> Client/server architecture
> In the client/server architecture (Figure 6.9), a subsystem, the server, provides servicesto instances
> of other subsystems called the clients, which are responsible for interacting with theuser. The request for a service is usually done via a remote procedure call mechanism or acommon object broker (e.g., CORBA or Java RMI). Control flow in the clients and the servers isindependent except for synchronization to manage requests or to receive results.
> Figure 6.9 Client/server architecture (UML class diagram).
> Clients request services from one or more servers. The server has no knowledge of the client. The client/server architecture is a generalization of the repository architecture.

Software Engineering module | PDF page 119 | lines 13-25

> 6.1.2. Services and subsystem interfaces
> A subsystem is characterized by the services it provides to other subsystems. A service is a set of

related operations that share a common purpose. A subsystem providing a notification service, for
example, defines operations to send notices, look up notification channels, and subscribe and unsubscribe to a channel.
The set of operations of a subsystem that are available to other subsystems form the subsystem interface. The subsystem interface, also referred to as the application programmer interface (API),
includes the name of the operations, their parameters, their types, and their return values. System
design focuses on defining the services provided by each subsystem that is, enumerating the operations, their parameters, and their high-level behavior. Object design will focus on defining the subsystem interfaces, that is, the type of the parameters and the return value of each operation.
The definition of a subsystem in terms of the services it provides helps us focus on its interface as opposed to its implementation. A good subsystem interface should provide as little information

Question sources

Computer Science MCQ practice collection · Question 20

Question 185 | AM0021

Which phase of the software development life cycle (SDLC) focuses on gathering and understanding user requirements?

A. Analysis

B. Design

C. Implementation

D. Testing

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer A: Analysis

Elicitation and analysis establish user needs before internal design and implementation.

Why each choice is right or wrong

A - Correct: Corresponds to gathering and structuring requirements. See PDF p. 62, 100.

B - Incorrect: Plans the internal realization. See PDF p. 62, 100.

C - Incorrect: Realizes the software design in code. See PDF p. 62, 100.

D - Incorrect: Tries to expose faults in implemented software. See PDF p. 62, 100.

Primary note proof

Study-note section SE08

Business requirements explain why the project exists and its goals. User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints. The Software Requirements Specification (SRS) organizes the agreed requirements. A natural-language specification communicates with clients while analysis models formalize it for technical discussion; real specifications can also contain diagrams.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 62 | lines 9-16

Requirement's elicitation focuses on describing the purpose of the system. The client, the developers, and the users identify a problem area and define a system that addresses the problem.
Such a definition is called a system specification and serves as a contract between the client and the developers. The system specification is structured and formalized during analysis to produce an analysis model (see Figure 3.1). Both system specification and analysis model represent the same information. They differ only in the language and notation they use. The system specification
is written in natural language, whereas the analysis model is usually expressed in a formal or semiformal notation.

Software Engineering module | PDF page 100 | lines 11-24

Analysis focuses on producing a model of the system, called the analysis model, which is correct, complete, consistent, and verifiable. Analysis is different from requirements elicitation in that developers focus on structuring and formalizing the requirements elicited from users
Figure 4.1 Products of requirements elicitation and analysis (UML activity diagram).
Althoughtheanalysismodelmaynotbeunderstandabletotheusersandtheclient, it helps developers verify the system specification produced during requirements elicitation. There is a natural tendency for users and developers to postpone difficult decisions until later in the project. A decision may be difficult because of lack of domain knowledge, lack of technological knowledge, or simply because of disagreements among users and developers. Postponing decisions enables the
project to move on smoothly and avoid confrontation with reality or peers. Unfortunately, difficult
decisions will eventually need to be made, often at higher cost when intrinsic problems are discovered during testing, or worse, during user evaluation. Translating a system specification in to a formal or semi-formal model forces developers to identify and resolve difficult issues early in
the development.

Question sources

Computer Science MCQ practice collection · Question 21

Question 186 | AM0022

What is the primary goal of software design?

A. To create a visually appealing user interface

B. To optimize the performance of the software system

C. To translate user requirements into a detailed plan

D. To develop and test individual software components

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: To translate user requirements into a detailed plan

Design transforms the SRS into a structure suitable for implementation and identifies internal architecture and responsibilities.

Why each choice is right or wrong

A - Incorrect: UI appearance is one design concern, not the whole goal. See PDF p. 18, 116.

B - Incorrect: Performance optimization is a design goal, not the full transformation purpose. See PDF p. 18, 116.

C - Correct: Matches transforming requirements into an implementable design. See PDF p. 18, 116.

D - Incorrect: Coding/testing occur after or alongside design refinement, not its primary deliverable. See PDF p. 18, 116.

Primary note proof

Study-note section SE21

> System design turns the analysis model into architecture: design goals, subsystem responsibilities, dependencies, hardware mapping, data management, access control, control flow and startup/shutdown/error policies. Detail/object design refines APIs, parameter/return types, exceptions, component choices and algorithms. A subsystem is a manageable logical collection with services and an interface; it need not be a physical replaceable artifact.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 18 | lines 13-20

> System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
> Software design involves identifying and describing the fundamental software system abstractions
> and their relationships. The goal of the design phase is to transform the requirements specified in
> the SRS document into a structure that is suitable for implementation in some programming

language. In technical terms, during the design phase the software architecture is derived from the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Software Engineering module | PDF page 116 | lines 16-29

The analysis model describes the system completely from the actors' point of view and serves as the basis of communication between the client and the developers. The analysis model, however, does not contain information about the internal structure of the system, its hardware configuration,
or, more generally, how the system should be realized. System design is the first step in this direction. System design results in the following products:
• List of design goals, describing the qualities of the system that developers should optimize
• Software architecture, describing the subsystem decomposition in terms of subsystem responsibilities, dependencies among subsystems, subsystem mapping to hardware, and major policy decisions such as control flow, access control, and data storage.
The design goals are derived from the nonfunctional requirements. Design goals guide the decisions to be made by the developers especially when trade-offs are needed. The subsystem decomposition constitutes the bulk of system design. Developers divide the system into manageable pieces to deal with complexity: Each subsystem is assigned to a team and realized independently. In order for this to be possible, though, developers need to address system-wide

Question sources

Computer Science MCQ practice collection · Question 22

Question 187 | AM0023

Which broad approach explicitly centers its values on collaboration, working results and responding to change, using frequent feedback to improve later work?

A. Waterfall model

B. Agile methodology

C. Spiral model

D. V-model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Agile methodology

The stated priorities combine short feedback cycles, collaboration and adaptation as central development values. Agile names that broad approach. Waterfall and the V-model emphasize planned stages, with the V-model additionally relating stages to test levels. Spiral can also iterate and consult stakeholders, but its defining cycle is risk-driven. Distinguishing the core emphasis avoids incorrectly claiming that only Agile can communicate or revisit work.

Why each choice is right or wrong

A - Incorrect: Classical waterfall emphasizes planned phase commitments.

B - Correct: Agile makes collaboration and responsiveness to change central values.

C - Incorrect: Spiral is defined by explicit risk-reduction cycles rather than this value-based emphasis.

D - Incorrect: The V-model is defined by development-test correspondences.

Primary note proof

Study-note section SE05

> Agile favors people collaborating, useful working results, customer participation and adaptation over rigid adherence to procedures, exhaustive documents, contracts or fixed plans. The latter still have value. Agile does not mean no planning or no documentation. A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

> Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to

follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Software Engineering module | PDF page 22 | lines 8-12

This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
also provide a platform for evaluation by the user.

Question sources

Computer Science MCQ practice collection · Question 23

Question 188 | AM0024

A team separates a reusable reporting component behind a stable interface so other applications can use it and its implementation can change without rewriting clients. Which benefit most directly captures both goals?

A. To improve the readability of the code

B. To ensure efficient memory management

C. To facilitate code reuse and maintenance

D. To minimize the number of software defects

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: To facilitate code reuse and maintenance

A module with a stable interface can be reused in another application, and hiding its implementation allows localized changes without rewriting its clients. Those are reuse and maintenance benefits. Readability may improve as well, but it does not fully describe the two goals stated in the scenario. Modularization does not automatically optimize memory use or eliminate defects; good boundaries and verification still matter.

Why each choice is right or wrong

A - Incorrect: Readability may improve, but it does not fully describe reuse across applications and localized change.

B - Incorrect: Memory efficiency is not guaranteed by separating modules.

C - Correct: Reuse and maintenance directly match the two stated goals.

D - Incorrect: Reducing defects can be a benefit, but it does not name the stated reuse-and-change objective.

Primary note proof

Study-note section SE22

Logical cohesion groups similar-category operations, often selected by a flag; temporal cohesion groups operations done at the same time; functional cohesion focuses on one task. Splitting an unrelated three-purpose component improves modularity. If behavior is preserved, the restructuring is refactoring; the words are not mutually exclusive answers. Smaller modules can aid understanding, independent work, reuse and maintenance, but too many interfaces also create complexity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 118 | lines 11-16

6.1.1. Subsystems and classes
In order to reduce the complexity of the application domain, we identified smaller parts called classes and organized them into packages. Similarly, to reduce the complexity of the solution domain, we decompose a system into simpler parts, called subsystems, which are made of a number of solution domain classes. In the case of complex subsystems, we recursively apply this principle and decompose a subsystem into simpler subsystems (see Figure 6.2).

Software Engineering module | PDF page 32 | lines 12-21

A class in an object-oriented system carefully delineates between its interfaces the routines and attributes within a class are held together tightly. In a properly designed system, the classes will be grouped into subsystems but remain independent; therefore, changing one class has no impact
on other classes, and so, the impact is minimized. However, the object-oriented approach is not a panacea; nothing is magical here that will promote perfect design or perfect code.
Promotion of reusability
Objects are reusable because they are modeled directly out of a real-world problem domain. Each
object stands by itself or within a small circle of peers (other objects). Within this framework, the class does not concern itself with the rest of the system or how it is going to be used within a particular system.

Software Engineering module | PDF page 120 | lines 6-12

6.1.3. Coupling and coherence
Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Question sources

Computer Science MCQ practice collection · Question 24

Question 189 | AM0025

Which type of testing aims to uncover defects in the interaction between different components of a software system?

A. Unit testing

B. Integration testing

C. System testing

D. Regression testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: Integration testing

The defects sought are in communication or interfaces among combined components, so the applicable level is integration testing. Unit tests concentrate on isolated units and system tests assess the complete product. Regression testing instead describes checking for unintended effects after a change, and can include unit, integration or system tests. The distinction is between the scope of combination and the purpose that triggers a test run.

Why each choice is right or wrong

A - Incorrect: Unit testing concentrates on an isolated function or small component.

B - Correct: Integration testing targets interactions among combined parts.

C - Incorrect: System testing checks the whole integrated system against requirements.

D - Incorrect: Regression is an after-change objective that can occur at several testing levels.

Primary note proof

Study-note section SE35

> Unit testing isolates a function, class or small component, often replacing dependencies with stubs/mocks; a driver invokes the unit and a stub substitutes a called dependency. Integration testing exercises interfaces and interactions among combined parts. System testing checks the complete integrated system against functional and nonfunctional requirements in a representative environment.

Study-note section SE36

> Functional tests check required behavior against the SRS and acceptance criteria. Performance tests assess speed, throughput and resources under a specified workload; load uses expected demands and stress exceeds normal limits. Usability, security, compatibility and reliability tests

target their respective qualities. “Normal operating conditions” alone does not distinguish functional from performance testing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Question sources

Computer Science MCQ practice collection · Question 25

Question 190 | AM0026

Which programming paradigm is based on the concept of objects and their interactions?

A. Functional programming

B. Procedural programming

C. Object-oriented programming

D. Event-driven programming

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Object-oriented programming

Object-oriented programming organizes a system around objects that combine state and behavior and interact through operations. Procedural programming decomposes work into functions or procedures that manipulate data. Functional programming emphasizes function evaluation and controlled side effects, while event-driven programming organizes reactions to events. An object-oriented program may also use events or functions, but the question asks which paradigm is defined by objects and their interactions.

Why each choice is right or wrong

A - Incorrect: Functional programming emphasizes evaluation of functions and controlled side effects.

B - Incorrect: Procedural programming emphasizes functions or procedures operating on data.

C - Correct: Object-oriented programming centers on objects with state and behavior.

D - Incorrect: Event-driven programming centers on reactions to events, independent of object-based decomposition.

Primary note proof

Study-note section SE01

> Procedural decomposition organizes functions; object oriented decomposition organizes objects with state and behavior. Either can use iterative or sequential development. OOP does not automatically shorten every project. Reuse, encapsulation and consistent models can reduce duplication and ease change; poor object design can still be complex. Functional programming emphasizes function evaluation and controlled side effects; event-driven programs react to events.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 40 | lines 12-18

> ✓ Object -Oriented paradigm is an approach to the solution of problems in which all computations are performed in the context of objects.
> ✓ Object-oriented programming allows decomposition of a problem into a number of objects and then builds data and functions around these objects.
> ✓ In an object-oriented environment everything is an object and each object is responsible for itself.
> ✓ Object oriented systems are easier to adapt to changes, more robust, and easier to maintain.

Question sources

Computer Science MCQ practice collection · Question 26

Question 191 | AM0027

What is the defining artifact-management function of version control, distinguishing it from collaboration benefits and CI services?

A. To manage project schedules and milestones

B. To facilitate collaboration among team members

C. To track and control changes to source code and files

D. To automate the testing and deployment processes

Hint

Separate artifact history and controlled changes from scheduling, collaboration benefits and automated execution.

Answer C: To track and control changes to source code and files

Version control records artifact versions and their history, making it possible to inspect changes, maintain branches and combine compatible work. Collaboration is an important benefit, but it is broader than the defining version-history function. Scheduling belongs to project planning, while automated testing and deployment belong to CI or delivery tooling that can use version-control events. These roles can be integrated without becoming the same activity.

Why each choice is right or wrong

A - Incorrect: Scheduling and milestones belong to project planning and tracking.

B - Incorrect: Collaboration is a benefit enabled by shared versions and merging, rather than the defining artifact-history function.

C - Correct: Tracking and controlling file changes is the direct version-control function.

D - Incorrect: Testing and deployment automation is CI/delivery work that may integrate with version control.

Primary note proof

Study-note section SE33

> Configuration management identifies controlled items, versions/baselines, change procedures, status and audits. Items may include source code, requirements, manuals, test plans, environments and build configurations. A team’s ordinary browser is usually a tool rather than product CI, but a controlled test-browser version may be an environment CI.

Study-note section SE28

> Git tracks versions; issue trackers such as Jira track work/defects; CI tools such as Jenkins automate checks; Docker packages application/runtime dependencies into containers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 142 | lines 5-10

Configuration management avoids faults caused by undisciplined change in the system models. For example, it is a common mistake to a change a subsystem interface without notifying all developers of calling components. Configuration management can make sure that, if analysis models and code are becoming inconsistent with one another, analysts and implementor are notified. The assumption behind configuration management is that the system contains many less
faults if change is controlled.

Question sources

Computer Science MCQ practice collection · Question 27

Question 192 | AM0028

Which conventional lifecycle is described by a simple linear requirements→design→implementation→testing sequence with reviewed phase completion, without the V-model's explicit development-to-test pairing?

A. Waterfall model

B. Agile methodology

C. Spiral model

D. V-model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A: Waterfall model

Classical waterfall organizes work as separated planned phases with reviewed deliverables, matching the stated linear sequence. The V-model additionally pairs development artifacts with later test levels, which this question excludes. Agile emphasizes adaptation and feedback, while spiral organizes repeated risk-driven cycles. Explicitly distinguishing the test-pairing structure removes the overlap between two staged processes without denying their relationship.

Why each choice is right or wrong

A - Correct: Classical waterfall matches the specified simple linear organization.

B - Incorrect: Agile emphasizes adaptive content and feedback.

C - Incorrect: Spiral uses risk-driven cycles.

D - Incorrect: The V-model adds explicit development-to-test pairing excluded here.

Primary note proof

Study-note section SE03

The V-model pairs specification/design stages with later test levels. Requirements inform acceptance tests; system design informs system tests; component/detail design informs

integration/unit tests. Coding lies at the bottom. Test planning starts early. Stable requirements and strong assurance may favor this organization, but neither a V shape nor waterfall guarantees safety or correctness.

Study-note section SE03

Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 15 | lines 16-20

non-iterative (or sequential) process in which progress is seen as flowing steadily downwards (like
a waterfall) through the development phases. Thus the waterfall model maintains that one should
move to a phase only when its preceding phase is reviewed and verified. That is, each phase must
be completed before the next phase can begin. In waterfall model phases do not overlap. But there
are various modified waterfall models, and hence, may include slight or major modifications.

Question sources

Computer Science MCQ practice collection · Question 28

Question 193 | AM0029

What is the primary purpose of software documentation?

A. To serve as a record of code authorship

B. To provide instructions for end-users

C. To assist in software maintenance and understanding

D. To ensure compliance with coding standards

Hint

Distinguish process assurance from product evaluation, and compare specified obligations with actual stakeholder needs.

Answer C: To assist in software maintenance and understanding

Documentation records requirements and architecture so developers can understand decisions, revisit designs and coordinate later change. A user manual is one audience-specific part of this broader purpose.

Why each choice is right or wrong

A - Incorrect: Authorship alone does not capture system decisions or behavior. See PDF p. 130, 83, 32.

B - Incorrect: End-user instructions are a valid subset, not the overall technical purpose. See PDF p. 130, 83, 32.

C - Correct: Matches understanding, revisiting and maintaining documented structures. See PDF p. 130, 83, 32.

D - Incorrect: Coding-standard compliance is narrower than the general understanding purpose. See PDF p. 130, 83, 32.

Primary note proof

Study-note section SE34

Quality assurance establishes and checks processes/standards; quality control evaluates work products. An SQA plan identifies standards, responsibilities, evaluations, reviews, audits and assurance records. Quality of conformance means following the design/specification; a product can conform to a poor specification. Verification asks whether specified obligations are implemented correctly; validation asks whether the product meets intended stakeholder needs. Both can use reviews, analysis or execution, so “human-only versus computer-only” is false.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 130 | lines 3-11

6.3. DOCUMENTING SYSTEM DESIGN
System design is documented in the System Design Document (SDD). It describes design goals set by the project, the subsystem decomposition (with UML class diagrams), the hardware/software mapping (with UML deployment diagrams), the data management, the access control, control flow mechanisms, and boundary conditions. The SDD is used to define interfaces between teams of developers and as reference when architecture-level decisions need to be revisited. The audience for the SDD includes the project management, the system architects (i.e., the developers who participate in the system design), and the developers who design and implement each subsystem. The following is an example template for a SDD:

Software Engineering module | PDF page 83 | lines 12-25

3.4.4. Documenting Requirements Elicitation
The results of the requirements elicitation activity and the analysis activity are documented in the
Requirements Analysis Document (RAD). This document completely describes the system in terms of functional and nonfunctional requirements and serves as a contractual basis between the
client and the developers. The audience for the RAD includes the client, the users, the project management, the system analysts (i.e., the developers who participate in the requirements), and the system designers (i.e., the developers who participate in the system design). The first part of the document, including use cases and nonfunctional requirements, is written during requirements

elicitation. The formalization of the specification in terms of object models is written during analysis.
The first section of the RAD is an Introduction. Its purpose is to provide a brief overview of the function of the system and the reasons for its development, its scope, and references to the development context (e.g., related problem statement, references to existing systems, feasibility studies). The introduction also includes the objectives and success criteria of the project.

Software Engineering module | PDF page 32 | lines 12-21

A class in an object-oriented system carefully delineates between its interfaces the routines and attributes within a class are held together tightly. In a properly designed system, the classes will be grouped into subsystems but remain independent; therefore, changing one class has no impact
on other classes, and so, the impact is minimized. However, the object-oriented approach is not a panacea; nothing is magical here that will promote perfect design or perfect code.
Promotion of reusability
Objects are reusable because they are modeled directly out of a real-world problem domain. Each
object stands by itself or within a small circle of peers (other objects). Within this framework, the class does not concern itself with the rest of the system or how it is going to be used within a particular system.

Question sources

Computer Science MCQ practice collection · Question 29

Question 194 | AM0030

After each short usable release, customers reorder the remaining work and the team adapts the next release’s content while keeping its timebox. Which broad approach most directly fits this planning behavior?

A. Waterfall model

B. Agile methodology

C. Spiral model

D. V-model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Agile methodology

A fixed short timebox with usable results and customer-driven adaptation of later content fits Agile planning. Classical waterfall and the V-model organize planned development stages, with the latter pairing stages to tests. Spiral cycles organize commitments around risk assessment and reduction. All can incorporate some staged work, so the stated treatment of feedback and later scope is what makes the Agile option the most direct match.

Why each choice is right or wrong

A - Incorrect: Classical waterfall emphasizes planned phases and later delivery feedback.

B - Correct: Agile adapts later work from customer feedback while producing usable increments.

C - Incorrect: Spiral is specifically organized around risk reduction, not the stated timeboxed adaptation rule.

D - Incorrect: The V-model is defined by paired development and test stages.

Primary note proof

Study-note section SE04

> Choose incremental delivery when prioritized partial capability must reach users early. Smaller changes can be easier to debug and permit feedback, though architecture and integration still require planning. Iteration need not rebuild the whole system every time, and increments need not be perfectly independent modules. Broad descriptions of early working software can fit prototypes, iterative or Agile processes; identify the purpose of that early version.

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Study-note section SE05

> Scrum uses timeboxed Sprints of at most one month, with a Sprint Goal, selected work and a usable Increment meeting a Definition of Done. Scope is discussed before and during planning; it is not forbidden to agree scope. Timeboxes tend to stay fixed while content is adapted without endangering the goal. Agile can be combined with required assurance/documentation; safety-critical context alone cannot prove it is unsuitable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 22 | lines 8-12

> This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
> also provide a platform for evaluation by the user.

Software Engineering module | PDF page 13 | lines 22-27

> Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to
> follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Question sources

Computer Science MCQ practice collection · Question 30

Question 195 | AM0031

Which design principle advocates for designing software components that have a single responsibility?

A. Encapsulation

B. Inheritance

C. Polymorphism

D. Single Responsibility Principle

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: Single Responsibility Principle.

SE25 identifies Single Responsibility among the core design principles. The question asks directly for one responsibility per component, so D is the exact named rule. Encapsulation protects internal decisions, inheritance creates subtype relations, and polymorphism supports varying implementations through a common interface. Those can help a design but do not themselves define a single responsibility.

Why each choice is right or wrong

A - Incorrect: Encapsulation hides implementation details rather than requiring one responsibility per component.

B - Incorrect: Inheritance establishes superclass/subclass reuse rather than the focused-responsibility rule.

C - Incorrect: Polymorphism supports compatible varying implementations rather than defining responsibility count.

D - Correct: Single Responsibility explicitly calls for one focused responsibility in a software component.

Primary note proof

Study-note section SE25

> SOLID includes Single Responsibility, Open-Closed, Liskov substitution, Interface Segregation and Dependency Inversion.

Study-note section SE-S02

> Generalization places common concepts in a more general superclass and specialized concepts in subclasses. Available inherited attributes and operations are shared by subclasses, which may extend or override behavior.

Study-note section SE-S02

> Encapsulation groups state with operations and hides representation behind an interface. Polymorphism permits different object types to respond differently through a common operation or interface. Abstraction exposes the relevant contract while omitting unnecessary detail. Reusability means using suitable existing classes or components in new contexts rather than writing them again. These ideas support each other but name different properties.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 120 | lines 24-27

> Coherence is the strength of dependencies within a subsystem. If a subsystem contains many objects that are related to each other and perform similar tasks, its coherence is high. If a subsystem
> contains a number of unrelated objects, its coherence is low. A desirable property of a subsystem decomposition is that it leads to subsystems with high coherence.

Software Engineering module | PDF page 101 | lines 9-15

> The analysis object model consists of entity, boundary, and control objects. Entity objects represent the persistent information tracked by the system. Boundary objects represent the interactions between the actors and the system. Control objects represent the tasks that a formed by the user and supported by the system. In the 2B watch example, Year, Month, Day are entity objects; Button Boundary and LCD Display Boundary are boundary objects; Change Date Control is
> a control object that represents the activity of changing the date by pressing combinations of buttons.

Question sources

Computer Science MCQ practice collection · Question 31

Question 196 | AM0032

Which design pattern is used to ensure that only one instance of a class is created throughout the system?

A. Singleton pattern

B. Factory pattern

C. Observer pattern

D. Decorator pattern

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer A: Singleton.

SE27 describes Singleton as controlling access to one instance. That is the pattern requested by the one-instance condition. Factory patterns address how concrete objects are created, Observer notifies subscribers and Decorator adds behavior by wrapping. None of those other pattern responsibilities guarantees a single shared instance by itself.

Why each choice is right or wrong

A - Correct: Singleton controls creation and access so that the intended class has one shared instance.

B - Incorrect: A factory selects or constructs products rather than inherently limiting a class to one instance.

C - Incorrect: Observer distributes notifications to subscribers rather than restricting object instance count.

D - Incorrect: Decorator adds behavior through wrapping rather than enforcing a single instance.

Primary note proof

Study-note section SE27

> Factory Method lets subclasses choose a concrete product through a creation operation, reducing coupling to concrete constructors. A simple factory centralizes creation but is not identical to Factory Method. Singleton controls access to one instance; Observer notifies subscribers; Memento stores/restores state while hiding representation; Iterator traverses; Decorator adds behavior through wrapping. The pattern name is Memento, not Momento.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 121 | lines 6-15

6.1.4. Software architecture
As the complexity of systems increases, the specification of the system decomposition is critical. It is difficult to modify or correct a weak decomposition once development has started, as most subsystem interfaces have to change. In recognition of the importance of this problem, the concept
of software architecture has emerged. A software architecture includes the system decomposition, the global control flow, error-handling policies and inter subsystem communication protocols.
In this section, we describe a few sample architectures that can be used for different systems. This
is by no means a systematic or thorough exposition of the subject. Rather, we aim to provide a few
representative examples and refer the reader to the literature for more details.

Question sources

Computer Science MCQ practice collection · Question 32

Question 197 | AM0033

Which test type checks that specified services produce the required outputs under normal operating conditions, without measuring speed/resources or user task ease?

A. Stress testing

B. Performance testing

C. Functional testing

D. Usability testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer C: Functional testing

Functional tests compare observable services and outputs with the SRS and acceptance criteria. That is the stated objective, so C applies. Performance testing measures speed, throughput or resources, and usability evaluates user task success or ease; neither is requested here. Stress testing exceeds normal limits. Normal operating conditions alone would not distinguish these types, but the explicit behavioral objective does.

Why each choice is right or wrong

A - Incorrect: Stress testing exceeds normal operating limits.

B - Incorrect: Performance testing measures speed, throughput or resources.

C - Correct: Functional testing checks specified services and outputs.

D - Incorrect: Usability testing evaluates user task ease or success.

Primary note proof

Study-note section SE36

Functional tests check required behavior against the SRS and acceptance criteria. Performance tests assess speed, throughput and resources under a specified workload; load uses expected demands and stress exceeds normal limits. Usability, security, compatibility and reliability tests target their respective qualities. “Normal operating conditions” alone does not distinguish functional from performance testing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Software Engineering module | PDF page 79 | lines 3-9

• Performance characteristics. How responsive should the system be? How many concurrent users should it support? What is a typical or extreme load?
• Error handling and extreme conditions. How should the system handle exceptions? Which exceptions should the system handle? What is the worse environment in which the system is expected to perform? Are there safety requirements on the system?
• Quality issues. How reliable/available/robust should the system be? What is the client’s involvement in assessing the quality of the system or the development process?

Question sources

Computer Science MCQ practice collection · Question 33

Question 198 | AM0034

Which programming paradigm promotes the use of data and avoids side effects?

A. Functional programming

B. Procedural programming

C. Object-oriented programming

D. Event-driven programming

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A: Functional programming, for the side-effect emphasis.

The phrase promotes the use of data is not distinctive because all these paradigms use data. The discriminating part is avoiding or controlling side effects. SE01 explicitly assigns function evaluation and controlled side effects to functional programming. Procedural, object-oriented and event-driven styles organize behavior differently and do not inherently impose that emphasis.

Why each choice is right or wrong

A - Correct: Functional programming emphasizes evaluating functions and controlling side effects.

B - Incorrect: Procedural decomposition organizes functions or procedures without inherently avoiding state-changing effects.

C - Incorrect: Object-oriented decomposition combines state and behavior and does not inherently exclude side effects.

D - Incorrect: Event-driven programs react to events; that does not itself require avoiding side effects.

Primary note proof

Study-note section SE01

> Procedural decomposition organizes functions; object oriented decomposition organizes objects with state and behavior. Either can use iterative or sequential development. OOP does not automatically shorten every project. Reuse, encapsulation and consistent models can reduce duplication and ease change; poor object design can still be complex. Functional programming emphasizes function evaluation and controlled side effects; event-driven programs react to events.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 40 | lines 12-18

✓ Object -Oriented paradigm is an approach to the solution of problems in which all computations are performed in the context of objects.
✓ Object-oriented programming allows decomposition of a problem into a number of objects and then builds data and functions around these objects.
✓ In an object-oriented environment everything is an object and each object is responsible for itself.
✓ Object oriented systems are easier to adapt to changes, more robust, and easier to maintain.

Question sources

Computer Science MCQ practice collection · Question 34

Question 199 | AM0035

Which type of software maintenance involves making changes to a software system to adapt it to new environments or technologies?

A. Corrective maintenance

B. Adaptive maintenance

C. Perfective maintenance

D. Preventive maintenance

Hint

Classify the motive for the change before its implementation: repair, environment, useful capability or reduced future risk.

Answer B: Adaptive maintenance.

SE40 defines adaptive maintenance as responding to changes in the environment. New platforms or technologies therefore create an adaptive-maintenance need even when the original software is not faulty. Corrective maintenance fixes faults, perfective work improves useful capability or performance, and preventive work reduces future risk. The stated environmental motive makes B the appropriate category.

Why each choice is right or wrong

A - Incorrect: Corrective work fixes faults rather than primarily accommodating changed environments or technology.

B - Correct: Adaptive maintenance changes software so it continues working in a new environment or technology context.

C - Incorrect: Perfective work improves capability or performance rather than specifically responding to an environmental change.

D - Incorrect: Preventive work reduces future maintenance risk rather than defining adaptation to a new environment.

Primary note proof

Study-note section SE40

> Corrective maintenance fixes faults; adaptive maintenance responds to environment changes; perfective maintenance improves useful capability/performance; preventive maintenance reduces future risk, such as restructuring fragile code. Motive determines category; refactoring can support preventive or perfective work. Maintenance often lasts years and includes understanding legacy code and regression effort, so it may dominate lifecycle cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 19 | lines 24-29

> 5. Operation and maintenance: Normally (although not necessarily), this is the longest life cycle phase. Maintenance of a typical software product requires much more than the effort necessary to
> develop the product itself. The system is installed and put into practical use. Maintenance involves
> correcting errors which were not discovered in earlier stages of the life cycle, improving the implementation of system units and enhancing the system ‘s services as new requirements are discovered.

Software Engineering module | PDF page 41 | lines 16-17

> ✓ Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous and the software must evolve to remain useful.

Question sources

Computer Science MCQ practice collection · Question 35

Question 200 | AM0036

What is the purpose of usability testing in software development?

A. To evaluate the performance of the software system under stress conditions

B. To assess the user-friendliness and ease of use of the software system

C. To verify the compliance of the software system with specified requirements

D. To identify and fix defects or issues in the software system

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer B: To assess the user-friendliness and ease of use of the software system

Usability testing examines ease of learning, task completion and user errors, with the goal of identifying and improving usability problems.

Why each choice is right or wrong

A - Incorrect: Stress performance is not the ease-of-use focus. See PDF p. 82.

B - Correct: Directly matches ease of use and learning. See PDF p. 82.

C - Incorrect: General requirements conformance is broader than this specific user-focused test. See PDF p. 82.

D - Incorrect: General fault fixing is broader; the test identifies user interaction difficulties rather than implementing all repairs. See PDF p. 82.

Primary note proof

Study-note section SE36

> Functional tests check required behavior against the SRS and acceptance criteria. Performance tests assess speed, throughput and resources under a specified workload; load uses expected demands and stress exceeds normal limits. Usability, security, compatibility and reliability tests target their respective qualities. “Normal operating conditions” alone does not distinguish functional from performance testing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 82 | lines 3-12

> though this type of approach could be used for any parameter, usability tests focus on usability parameters, such as the ease to learn the system, the time to accomplish a task, or the rate of errors
> a user makes when accomplishing a task.
> There are two important differences between classical experiments and usability tests. Whereas the classical experimental method is designed to confirm or refute a hypothesis, the goal of usability tests is to obtain qualitative information on how to fix usability problems and how to improve the system. Another difference is the rigor with which the experiments need to be performed. It has been shown that even a series of quick focused tests starting as early as requirements elicitation is extremely helpful. Nielsen uses the term “discount usability engineering” to indicate that a few usability tests are better than none at all.

Question sources

Computer Science MCQ practice collection · Question 36

Question 201 | AM0037

Which software development methodology involves iterative development cycles called sprints?

A. Waterfall model

B. Agile methodology

C. Spiral model

D. V-model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Agile methodology, specifically Scrum.

SE05 places Sprints in Scrum, an Agile framework. B is therefore the available category that contains the specific process described. Agile as a whole does not require every method to call its iterations Sprints. Waterfall, spiral and the V-model have other organizing principles, so none is the named Sprint-based method.

Why each choice is right or wrong

A - Incorrect: Classical waterfall uses planned phases rather than Scrum Sprints.

B - Correct: Scrum is an Agile framework whose iterative delivery cycles are called Sprints.

C - Incorrect: Spiral uses risk-driven cycles rather than specifically the Scrum Sprint mechanism.

D - Incorrect: The V-model pairs development stages with tests rather than organizing work into Sprints.

Primary note proof

Study-note section SE05

> Scrum uses timeboxed Sprints of at most one month, with a Sprint Goal, selected work and a usable Increment meeting a Definition of Done. Scope is discussed before and during planning; it is not forbidden to agree scope. Timeboxes tend to stay fixed while content is adapted without endangering the goal. Agile can be combined with required assurance/documentation; safety-critical context alone cannot prove it is unsuitable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

> Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to
> follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Question sources

Computer Science MCQ practice collection · Question 37

Question 202 | AM0038

What is the primary purpose of code refactoring?

A. To add new features to the software system

B. To optimize the performance of the software system

C. To improve the readability and maintainability of the code

D. To fix defects or issues in the software system

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: To improve the readability and maintainability of the code

Restructuring explicitly addresses maintainability, readability and understandability, while optimization separately addresses speed and memory. C matches refactoring's purpose of improving internal structure and maintainability.

Why each choice is right or wrong

A - Incorrect: Adds behavior, whereas restructuring reorganizes existing design elements. See PDF p. 133.

B - Incorrect: Optimization is separately classified by performance goals. See PDF p. 133.

C - Correct: Direct match to the documented restructuring goals. See PDF p. 133.

D - Incorrect: Repairing a fault is correction; restructuring need not fix a behavioral defect. See PDF p. 133.

Primary note proof

Study-note section SE43

> Pareto's 80/20 rule is a prioritization heuristic: a small subset of causes may account for much of the observed impact. It does not guarantee 80% of defects can be fixed with exactly 20% effort. Refactor to preserve behavior while improving structure; optimize only against measured needs; record rationale and update traceability and regression evidence after change.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 133 | lines 3-15

> transformation on subsets of a particular model. Typical activities include transforming n-ary associations into binary associations, implementing binary associations as references, merging two similar classes from two different subsystems into a single class, collapsing classes with no significant behavior into attributes, splitting complex classes into simpler ones, rearranging classes and operations to increase the inheritance and packaging. During restructuring, we address design goals such as maintainability, readability, and understandability of the system model.

• Optimization. Optimization activities address performance requirements of the system model. This includes changing algorithms to respond to speed or memory requirements, reducing multiplicities in associations to speed up queries, adding redundant associations for efficiency, rearranging execution orders, adding derived attributes to improve the access time to objects and opening up the architecture, that is, adding access to lower layers because of performance requirements.

Question sources

Computer Science MCQ practice collection · Question 38

Question 203 | AM0039

Which approach treats customer participation in reviewing working results and reprioritizing future work as a foundational value, rather than merely a practice a project may choose?

A. Waterfall model

B. Agile methodology

C. Spiral model

D. V-model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Agile methodology

The project treats customer participation in reviewing results and reprioritizing future work as a foundational development value. This directly matches Agile's customer collaboration and responsiveness to change. Waterfall and the V-model primarily define phase and test-stage organization. Spiral also includes stakeholder review, but risk assessment and reduction define its cycle. Participation alone is not exclusive; the explicit priority given to continual adaptive collaboration identifies the answer.

Why each choice is right or wrong

A - Incorrect: Classical waterfall's defining organization is planned phase progression.

B - Correct: Agile explicitly prioritizes customer collaboration and adapting to feedback.

C - Incorrect: Spiral includes review but is defined by risk-driven cycles.

D - Incorrect: The V-model organizes linked development and test stages, rather than naming the stated collaboration values.

Primary note proof

Study-note section SE05

Agile favors people collaborating, useful working results, customer participation and adaptation over rigid adherence to procedures, exhaustive documents, contracts or fixed plans. The latter still have value. Agile does not mean no planning or no documentation. A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Study-note section SE03

Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Study-note section SE06

Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 13 | lines 22-27

Sometimes, software processes are categorized as either plan-driven or agile processes. Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan. In agile processes planning is incremental and it is easier to change the process to reflect changing customer requirements. There is no hard and fast rules to
follow but each approach is suitable for different types of software. Generally, you need to find a balance between plan-driven and agile processes.

Software Engineering module | PDF page 22 | lines 8-12

This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
also provide a platform for evaluation by the user.

Software Engineering module | PDF page 81 | lines 11-21

3.4.2. Negotiating specifications with clients: Joint Application Design
Joint Application Design (JAD) is effectiveness lies in that the requirements elicitation work is done in one single workshop session in which all stakeholders participate. Users, clients, developers, and a trained session leader sit together in one room to present their viewpoint, listen
to other viewpoints, negotiate, and agree on a mutually acceptable solution. The outcome of the workshop, the final JAD document, is a complete system specification document that includes definitions of data elements, work flows, and interface screens. Because the final document is

jointly developed by the stakeholders (that is, the participants who not only have an interest in the
success of the project, but also can make substantial decisions) the final JAD document represents
an agreement between users, clients, and developers, and thus minimizes requirements changes later in the development process.

Question sources

Computer Science MCQ practice collection · Question 39

Question 204 | AM0040

Which type of testing involves running the entire software system in an environment that closely resembles the production environment?

A. Unit testing

B. Integration testing

C. System testing

D. Regression testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer C: System testing

System testing evaluates the complete integrated software against its functional and nonfunctional requirements in a representative environment. Unit testing isolates a smaller element and integration testing concentrates on interfaces among combined parts. Regression testing instead checks for unintended consequences after change and is not restricted to any one level or environment. The question specifies the complete-product scope, making system testing the direct classification.

Why each choice is right or wrong

A - Incorrect: Unit testing isolates a function, class or small component.

B - Incorrect: Integration testing focuses on interfaces and interactions among combined parts.

C - Correct: System testing evaluates the complete integrated product in a representative environment.

D - Incorrect: Regression names an after-change checking purpose and can run at multiple levels.

Primary note proof

Study-note section SE35

Unit testing isolates a function, class or small component, often replacing dependencies with stubs/mocks; a driver invokes the unit and a stub substitutes a called dependency. Integration testing exercises interfaces and interactions among combined parts. System testing checks the complete integrated system against functional and nonfunctional requirements in a representative environment.

Study-note section SE36

Functional tests check required behavior against the SRS and acceptance criteria. Performance tests assess speed, throughput and resources under a specified workload; load uses expected demands and stress exceeds normal limits. Usability, security, compatibility and reliability tests target their respective qualities. “Normal operating conditions” alone does not distinguish functional from performance testing.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 147 | lines 14-18

• Unit testing, which finds faults by isolating an individual component using test stubs and drivers and by exercising the component using a test case
• Integration testing, which finds faults by integrating several components together
• System testing, which focuses on the complete system, its functional and nonfunctional requirements, and its target environment

Software Engineering module | PDF page 144 | lines 17-32

Figure 6.2 depicts an overview of the testing activities:
• Unit testing tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Question sources

Computer Science MCQ practice collection · Question 40

Question 205 | AM0041

What is a use case in use case modeling?

A. A description of a specific user interface design

B. A step-by-step procedure for software installation

C. A sequence of activities performed by a user or system

D. A diagram that represents the relationships between software components

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: A sequence of activities performed by a user or system

A use case describes a system-provided function as a flow of events producing an actor-visible result. C is acceptable when understood as goal-related interactions, not any arbitrary sequence of activities.

Why each choice is right or wrong

A - Incorrect: UI layout is an implementation/interface design artifact. See PDF p. 47, 48.

B - Incorrect: Installation instructions are one possible documented procedure, not the general use-case definition. See PDF p. 47, 48.

C - Correct: Best matches the goal-oriented flow of events involving actors and the system. See PDF p. 47, 48.

D - Incorrect: Component relationships describe internal structure rather than actor-visible behavior. See PDF p. 47, 48.

Primary note proof

Study-note section SE14

> Draw actors outside the system boundary rectangle and use-case ovals inside. Name a goal with a verb and noun. The diagram shows roles, goals and relationships, not the full detailed algorithm or UI layout. Being an internal component is different from being an external actor; the chosen boundary determines which role applies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 47 | lines 30-39

> Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
> Use cases describe the behavior of the system as seen from an actor's point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate

a
use case to access the system functionality. The use case can then initiate other use cases and gather
more information from the actors. When actors and use cases exchange information, they are said
to communicate. We will see later that we represent these exchanges with communication relationships.

Software Engineering module | PDF page 48 | lines 18-32

To describe a use case, we use a template composed of six fields (see also Figure 2.2):
• The name of the use case is unique across the system so that developers (and project participants) can unambiguously refer to the use case.
• Participating actors are actors interacting with the use case.
• Entry conditions describe the conditions that need to be satisfied before the use case is initiated.
• The flow of events describes the sequence of actions of the use case, which are be numbered for reference. The common case (i.e., cases that occur frequently) and the exceptional cases (i.e., cases that seldom occur, such as errors and unusual conditions) are described separately in different use cases for clarity.
• Exit conditions describe the conditions that are satisfied after the completion of the use case.
• Special requirements are requirements that are not related to the functionality of the system. These include constraints on the performance of the system, its implementation, the hardware platforms it runs on, and so on.

Question sources

Computer Science MCQ practice collection · Question 41

Question 206 | AM0042

What is an actor in use-case modeling when the system boundary is fixed?

A. A component that performs specific actions in a software system

B. An external interacting role played by a person, device or another system.

C. A diagram that represents the flow of activities in a use case

D. A particular individual user account rather than a role.

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: An external interacting role played by a person, device or another system.

An actor is an external role that interacts across the chosen system boundary. A person, device or another system may play that role. It is not an internal component, a diagram, or merely one individual user account. Because actors model roles rather than particular people, one person can play multiple actors and multiple people can fulfill the same actor role.

Why each choice is right or wrong

A - Incorrect: An internal component is not an external actor merely because it acts.

B - Correct: The external interacting role is the actor.

C - Incorrect: An activity-flow diagram is a model artifact rather than an actor.

D - Incorrect: An individual account is not the role abstraction.

Primary note proof

Study-note section SE14

> Draw actors outside the system boundary rectangle and use-case ovals inside. Name a goal with a verb and noun. The diagram shows roles, goals and relationships, not the full detailed algorithm or UI layout. Being an internal component is different from being an external actor; the chosen boundary determines which role applies.

Study-note section SE14

> An actor is an external role interacting with the system; it can be a person, device or another system. A person may play several roles, and several people may play one role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 47 | lines 30-39

> Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
> Use cases describe the behavior of the system as seen from an actor’s point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate a
> use case to access the system functionality. The use case can then initiate other use cases and gather
> more information from the actors. When actors and use cases exchange information, they are said
> to communicate. We will see later that we represent these exchanges with communication relationships.

Question sources

Computer Science MCQ practice collection · Question 42

Question 207 | AM0043

What is the purpose of an include relationship between use cases?

A. To represent the reuse of common steps between use cases

B. To represent optional steps within a use case

C. To represent the extension of a use case with additional steps

D. To represent the relationship between actors and use cases

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer A: To represent the reuse of common steps between use cases

Factoring common functionality into one included use case eliminates redundant descriptions, as in shared HelpDispatcher behavior.

Why each choice is right or wrong

A - Correct: Direct reuse/commonality mechanism. See PDF p. 50, 50.

B - Incorrect: Optional exceptional behavior is more directly represented by extend. See PDF p. 50, 50.

C - Incorrect: Adds events under conditions through extend rather than reuse via include. See PDF p. 50, 50.

D - Incorrect: Actor/use-case communication uses associations, not include. See PDF p. 50, 50.

Primary note proof

Study-note section SE16

> Include factors required reused behavior: Submit Order includes Validate Order, with the dashed arrow from the including/base use case to the included one. Extend adds conditional behavior at extension points: Apply Discount extends Checkout, with the arrow from the extension to the base. The base is meaningful independently. For an extension, draw the arrow from the extending use case to the base use case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 50 | lines 3-8

> can be modeled by a use case HelpDispatcher that is included by the use cases OpenIncident and AllocateResources (and any other use cases accessible by the Dispatcher). The resulting model only describes the HelpDispatcher functionality once, thus reducing complexity. Two use cases are related by an include relationship if one of them includes the second one in its flow of events. In UML, include relationships are depicted by a dashed arrow originating from the including use case. Include relationships are labeled with the string <<include>>.

Software Engineering module | PDF page 50 | lines 14-24

Extend relationships
Extend relationships are an alternate means for reducing complexity in the use case model. A use
case can extend another use case by adding events. An extend relationship indicates that an instance of an extended use case may include (under certain conditions) the behavior specified by
the extending use case. A typical application of extend relationships is the specification of exceptional behavior. For example (Figure 2-19), assume that the network connection between the
Dispatcher and the FieldOfficer can be interrupted at any time. (e.g., if the FieldOfficer enters a tunnel). The use case ConnectionDown describes the set of events taken by the system and the actors while the connection is lost. ConnectionDown extends the use cases OpenIncident and AllocateResources. Separating exceptional behavior from common behavior enables us to write shorter and more focused use cases.

Question sources

Computer Science MCQ practice collection · Question 43

Question 208 | AM0044

Which statements describe extend between use cases? Select all that apply. Read optional as conditional behavior at an extension point.

A. To factor required reused steps into an included use case.

B. To represent optional steps within a use case

C. To represent the extension of a use case with additional steps

D. To represent the relationship between actors and use cases

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B, C: To represent optional steps within a use case; To represent the extension of a use case with additional steps

Extend adds conditional behavior at extension points in an independently meaningful base use case. It therefore represents optional additional behavior and extends the base with added steps, making both B and C true. Factoring required reused behavior is the include relationship. Associations between actors and use cases are separate from the extend relationship linking use cases to one another.

Why each choice is right or wrong

A - Incorrect: Required reused behavior is factored with include.

B - Correct: Conditional additional behavior need not occur in every base execution.

C - Correct: Extend adds steps to an independently meaningful base.

D - Incorrect: Actor association is a different relationship.

Primary note proof

Study-note section SE16

> Include factors required reused behavior: Submit Order includes Validate Order, with the dashed arrow from the including/base use case to the included one. Extend adds conditional behavior at extension points: Apply Discount extends Checkout, with the arrow from the extension to the base. The base is meaningful independently. For an extension, draw the arrow from the extending use case to the base use case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 50 | lines 14-24

> Extend relationships
> Extend relationships are an alternate means for reducing complexity in the use case model. A use
> case can extend another use case by adding events. An extend relationship indicates that an instance of an extended use case may include (under certain conditions) the behavior specified by
> the extending use case. A typical application of extend relationships is the specification of exceptional behavior. For example (Figure 2-19), assume that the network connection between the
> Dispatcher and the FieldOfficer can be interrupted at any time. (e.g., if the FieldOfficer enters a tunnel). The use case ConnectionDown describes the set of events taken by the system and the actors while the connection is lost. ConnectionDown extends the use cases OpenIncident and AllocateResources. Separating exceptional behavior from common behavior enables us to write shorter and more focused use cases.

Software Engineering module | PDF page 50 | lines 3-8

> can be modeled by a use case HelpDispatcher that is included by the use cases OpenIncident and AllocateResources (and any other use cases accessible by the Dispatcher). The resulting model only describes the HelpDispatcher functionality once, thus reducing complexity. Two use cases are related by an include relationship if one of them includes the second one in its flow of events. In UML, include relationships are depicted by a dashed arrow originating from the including use case. Include relationships are labeled with the string <<include>>.

Question sources

Computer Science MCQ practice collection · Question 44

Question 209 | AM0045

Which diagram is commonly used to visualize the relationships between actors and use cases in use case modeling?

A. Class diagram

B. Use case diagram

C. Activity diagram

D. Sequence diagram

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: Use case diagram

Use-case diagrams depict actors and use cases with communication associations; these are the requested external interaction relationships.

Why each choice is right or wrong

A - Incorrect: Describes class structure. See PDF p. 47, 48.

B - Correct: Direct actor/use-case representation. See PDF p. 47, 48.

C - Incorrect: Describes action/control flow. See PDF p. 47, 48.

D - Incorrect: Describes time-ordered object interactions, not the high-level actor/function map. See PDF p. 47, 48.

Primary note proof

Study-note section SE14

> Draw actors outside the system boundary rectangle and use-case ovals inside. Name a goal with a verb and noun. The diagram shows roles, goals and relationships, not the full detailed algorithm or UI layout. Being an internal component is different from being an external actor; the chosen boundary determines which role applies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 47 | lines 30-39

> Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
> Use cases describe the behavior of the system as seen from an actor's point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate a
> use case to access the system functionality. The use case can then initiate other use cases and

gather
more information from the actors. When actors and use cases exchange information, they are said
to communicate. We will see later that we represent these exchanges with communication relationships.

Software Engineering module | PDF page 48 | lines 18-32

To describe a use case, we use a template composed of six fields (see also Figure 2.2):
• The name of the use case is unique across the system so that developers (and project participants) can unambiguously refer to the use case.
• Participating actors are actors interacting with the use case.
• Entry conditions describe the conditions that need to be satisfied before the use case is initiated.
• The flow of events describes the sequence of actions of the use case, which are be numbered for reference. The common case (i.e., cases that occur frequently) and the exceptional cases (i.e., cases that seldom occur, such as errors and unusual conditions) are described separately in different use cases for clarity.
• Exit conditions describe the conditions that are satisfied after the completion of the use case.
• Special requirements are requirements that are not related to the functionality of the system. These include constraints on the performance of the system, its implementation, the hardware platforms it runs on, and so on.

Question sources

Computer Science MCQ practice collection · Question 45

Question 210 | AM0046

What defines an abstract use case, rather than optionality, unfinished documentation or reuse alone?

A. To represent a use case that is not yet fully defined

B. To represent a use case that is optional in the system

C. It specifies behavior for specialization and is not directly instantiated as a complete use case.

D. To represent a use case that is reused by other use cases

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: It specifies behavior for specialization and is not directly instantiated as a complete use case.

An abstract use case specifies behavior for specialization and is not directly instantiated as a complete use case. That is different from being unfinished, optional or reused. Concrete behavior can be included for reuse, and extend can express conditional additions. Nor does the absence of a direct actor line by itself establish abstraction. Option C states the actual defining criterion.

Why each choice is right or wrong

A - Incorrect: An incomplete description is not the abstraction criterion.

B - Incorrect: Optional behavior is not the abstraction criterion.

C - Correct: Specialization without direct complete instantiation defines this abstraction.

D - Incorrect: Reuse alone can also involve concrete use cases.

Primary note proof

Study-note section SE16

> Generalization relates a specialized actor/use case to a more general one, with a hollow triangle toward the general element. An abstract use case specifies behavior for specialization and is not directly instantiated as a complete use case. It is not defined merely by reuse or lack of a direct actor line. Included behavior is incorporated where the inclusion is reached; do not infer every conditional branch in the base always executes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 51 | lines 4-13

> Generalization relationships
> Generalization/specialization relationships are a third mechanism for reducing the complexity of a model. A use case can specialize a more general one by adding more detail. For example, FieldOfficers are required to authenticate before they can use FRIEND. During early stages of requirements elicitation, authentication is modeled as a high-level Authenticate use case. Later, developers describe the Authenticate use case in more detail and allow for several different hardware platforms. This refinement activity results in two more use cases, AuthenticateWithPasswordwhich enables FieldOfficers to login without any specific hardware, and AuthenticateWithCard, which enables FieldOfficers to login using a smart card. The two new use cases are represented as specializations of the Authenticate use case.

Software Engineering module | PDF page 50 | lines 3-8

> can be modeled by a use case HelpDispatcher that is included by the use cases OpenIncident and AllocateResources (and any other use cases accessible by the Dispatcher). The resulting model only describes the HelpDispatcher functionality once, thus reducing complexity. Two use cases are related by an include relationship if one of them includes the second one in its flow of events. In UML, include relationships are depicted by a dashed arrow originating from the including use case. Include relationships are labeled with the string <<include>>.

Question sources

Computer Science MCQ practice collection · Question 46

Question 211 | AM0047

Which option states the most specific defining purpose of an exception flow, distinguishing it from a generic alternative flow?

A. To represent an error or exceptional situation in the use case

B. To represent an alternative path within the use case

C. To represent the termination of the use case

D. To represent the relationship between primary and secondary actors

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer A: To represent an error or exceptional situation in the use case

An exception flow handles an error or failure encountered in the use case. Such a path is also an alternative to the main success flow in a broad sense, but B alone does not distinguish exceptional handling from normal conditional variations. An exception may allow correction and recovery instead of terminating the use case. Actor relationships do not define its failure-handling purpose.

Why each choice is right or wrong

A - Correct: Error or exceptional handling is the specific defining purpose.

B - Incorrect: Alternative path is broader and does not identify the exceptional nature.

C - Incorrect: Exceptions need not terminate the entire use case.

D - Incorrect: Actor relationships do not describe failure handling.

Primary note proof

Study-note section SE15

> Example: Submit Incident. Precondition: officer authenticated. 1 Officer supplies location and description. 2 System validates details. 3 System records the incident. 4 System returns its ID. Alternative 2a: location missing → report the problem and request correction; save nothing. Success postcondition: incident and ID persist; failure guarantee: no partial record. An alternate flow is a conditional variation; an exception flow handles a failure. Preconditions hold before entry, while a trigger starts the use case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 48 | lines 18-32

To describe a use case, we use a template composed of six fields (see also Figure 2.2):
• The name of the use case is unique across the system so that developers (and project participants) can unambiguously refer to the use case.
• Participating actors are actors interacting with the use case.
• Entry conditions describe the conditions that need to be satisfied before the use case is initiated.
• The flow of events describes the sequence of actions of the use case, which are be numbered for reference. The common case (i.e., cases that occur frequently) and the exceptional cases (i.e., cases that seldom occur, such as errors and unusual conditions) are described separately in different use cases for clarity.
• Exit conditions describe the conditions that are satisfied after the completion of the use case.
• Special requirements are requirements that are not related to the functionality of the system. These include constraints on the performance of the system, its implementation, the hardware platforms it runs on, and so on.

Software Engineering module | PDF page 50 | lines 14-24

Extend relationships
Extend relationships are an alternate means for reducing complexity in the use case model. A use
case can extend another use case by adding events. An extend relationship indicates that an instance of an extended use case may include (under certain conditions) the behavior specified by
the extending use case. A typical application of extend relationships is the specification of exceptional behavior. For example (Figure 2-19), assume that the network connection between the
Dispatcher and the FieldOfficer can be interrupted at any time. (e.g., if the FieldOfficer enters a tunnel). The use case ConnectionDown describes the set of events taken by the system and the actors while the connection is lost. ConnectionDown extends the use cases OpenIncident and AllocateResources. Separating exceptional behavior from common behavior enables us to write shorter and more focused use cases.

Question sources

Computer Science MCQ practice collection · Question 47

Question 212 | AM0048

What is the primary goal of use case modeling?

A. To define the software architecture of the system

B. To identify and document user requirements

C. To design the user interface of the system

D. To implement the functionality of the system

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: To identify and document user requirements

Use cases capture externally visible functionality and let developers validate requirements with users, rather than defining internal architecture or implementation.

Why each choice is right or wrong

A - Incorrect: System design defines architecture after the external analysis model. See PDF p. 47, 62.

B - Correct: Directly captures actor-visible functional requirements. See PDF p. 47, 62.

C - Incorrect: UI layout can be explored alongside use cases but is not their primary purpose. See PDF p. 47, 62.

D - Incorrect: Implementation realizes the requirements instead of documenting them. See PDF p. 47, 62.

Primary note proof

Study-note section SE14

> An actor is an external role interacting with the system; it can be a person, device or another system. A person may play several roles, and several people may play one role. A primary actor seeks a goal; a supporting actor provides a service. Ask which user groups perform main functions, administer or maintain the system, and which external systems exchange information.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 47 | lines 30-39

> Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
> Use cases describe the behavior of the system as seen from an actor's point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate a
> use case to access the system functionality. The use case can then initiate other use cases and gather
> more information from the actors. When actors and use cases exchange information, they are said
> to communicate. We will see later that we represent these exchanges with communication relationships.

Software Engineering module | PDF page 62 | lines 9-16

Requirement's elicitation focuses on describing the purpose of the system. The client, the developers, and the users identify a problem area and define a system that addresses the problem.
Such a definition is called a system specification and serves as a contract between the client and the developers. The system specification is structured and formalized during analysis to produce an analysis model (see Figure 3.1). Both system specification and analysis model represent the same information. They differ only in the language and notation they use. The system specification
is written in natural language, whereas the analysis model is usually expressed in a formal or semiformal notation.

Question sources

Computer Science MCQ practice collection · Question 48

Question 213 | AM0049

What is an alternative flow in a use case?

A. A flowchart representation of the use case steps

B. A flow of activities that occurs in parallel with the main flow

C. A different sequence of steps based on specific conditions

D. A representation of the relationship between use cases

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: A different sequence of steps based on specific conditions

A flow of events describes interactions; exceptional conditions introduce alternative event sequences. This is a different path under a condition, rather than necessarily concurrency or inter-use-case structure.

Why each choice is right or wrong

A - Incorrect: A flowchart is a representation format, not the alternative sequence itself. See PDF p. 48, 50.

B - Incorrect: Parallel execution is not required by an alternative branch. See PDF p. 48, 50.

C - Correct: Directly describes a condition-dependent event sequence. See PDF p. 48, 50.

D - Incorrect: Include/extend/generalization describe relationships, not the definition of every alternative flow. See PDF p. 48, 50.

Primary note proof

Study-note section SE15

Example: Submit Incident. Precondition: officer authenticated. 1 Officer supplies location and description. 2 System validates details. 3 System records the incident. 4 System returns its ID. Alternative 2a: location missing → report the problem and request correction; save nothing. Success postcondition: incident and ID persist; failure guarantee: no partial record. An alternate flow is a conditional variation; an exception flow handles a failure. Preconditions hold before entry, while a trigger starts the use case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 48 | lines 18-32

To describe a use case, we use a template composed of six fields (see also Figure 2.2):
• The name of the use case is unique across the system so that developers (and project participants) can unambiguously refer to the use case.
• Participating actors are actors interacting with the use case.
• Entry conditions describe the conditions that need to be satisfied before the use case is initiated.
• The flow of events describes the sequence of actions of the use case, which are be numbered for reference. The common case (i.e., cases that occur frequently) and the exceptional cases (i.e., cases that seldom occur, such as errors and unusual conditions) are described separately in different use cases for clarity.
• Exit conditions describe the conditions that are satisfied after the completion of the use case.
• Special requirements are requirements that are not related to the functionality of the system. These include constraints on the performance of the system, its implementation, the hardware platforms it runs on, and so on.

Software Engineering module | PDF page 50 | lines 14-24

Extend relationships
Extend relationships are an alternate means for reducing complexity in the use case model. A use
case can extend another use case by adding events. An extend relationship indicates that an instance of an extended use case may include (under certain conditions) the behavior specified by
the extending use case. A typical application of extend relationships is the specification of exceptional behavior. For example (Figure 2-19), assume that the network connection between the
Dispatcher and the FieldOfficer can be interrupted at any time. (e.g., if the FieldOfficer enters a tunnel). The use case ConnectionDown describes the set of events taken by the system and the actors while the connection is lost. ConnectionDown extends the use cases OpenIncident and AllocateResources. Separating exceptional behavior from common behavior enables us to write shorter and more focused use cases.

Question sources

Computer Science MCQ practice collection · Question 49

Question 214 | AM0050

What is the purpose of a use case diagram?

A. To describe the behavior of a use case in detail

B. To represent the internal structure of a use case

C. To visualize the relationships between actors and use cases

D. To represent the flow of activities within a use case

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: To visualize the relationships between actors and use cases

The diagram shows actors, functions and communication links; the separate textual template contains detailed event sequences and conditions.

Why each choice is right or wrong

A - Incorrect: Detailed flow belongs in the use-case description/template. See PDF p. 47, 48.

B - Incorrect: Internal structure is modeled with classes/components. See PDF p. 47, 48.

C - Correct: Direct high-level actor/function interaction view. See PDF p. 47, 48.

D - Incorrect: Activity diagrams or textual flows express detailed action sequences. See PDF p. 47, 48.

Primary note proof

Study-note section SE14

> Draw actors outside the system boundary rectangle and use-case ovals inside. Name a goal with a verb and noun. The diagram shows roles, goals and relationships, not the full detailed algorithm or UI layout. Being an internal component is different from being an external actor; the chosen boundary determines which role applies.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 47 | lines 30-39

> Actors are external entities that interact with the system. Examples of actors include a user role (e.g., a system administrator, a bank customer, a bank teller) or another system (e.g., a central database, a fabrication line). Actors have unique names and descriptions.
> Use cases describe the behavior of the system as seen from an actor's point of view. Behavior described by the use case model is also called external behavior. A use case describes a function provided by the system as a set of events that yields a visible result for the actors. Actors initiate a
> use case to access the system functionality. The use case can then initiate other use cases and gather

more information from the actors. When actors and use cases exchange information, they are said
to communicate. We will see later that we represent these exchanges with communication relationships.

Software Engineering module | PDF page 48 | lines 18-32

To describe a use case, we use a template composed of six fields (see also Figure 2.2):
• The name of the use case is unique across the system so that developers (and project participants) can unambiguously refer to the use case.
• Participating actors are actors interacting with the use case.
• Entry conditions describe the conditions that need to be satisfied before the use case is initiated.
• The flow of events describes the sequence of actions of the use case, which are be numbered for reference. The common case (i.e., cases that occur frequently) and the exceptional cases (i.e., cases that seldom occur, such as errors and unusual conditions) are described separately in different use cases for clarity.
• Exit conditions describe the conditions that are satisfied after the completion of the use case.
• Special requirements are requirements that are not related to the functionality of the system. These include constraints on the performance of the system, its implementation, the hardware platforms it runs on, and so on.

Question sources

Computer Science MCQ practice collection · Question 50

Question 215 | CS0007

Identifying and describing the fundamental software system abstractions and their relationships. Establishing the overall system architecture - in which phase are these activities done?

A. System design

B. Requirement analysis and definition

C. System testing

D. Implementation

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer A: System design

System and software design establish the overall architecture and identify software abstractions and relationships.

Why each choice is right or wrong

A - Correct: Direct match to the described design tasks. See PDF p. 18, 116.

B - Incorrect: Establishes required services, constraints and goals. See PDF p. 18, 116.

C - Incorrect: Checks complete-system behavior. See PDF p. 18, 116.

D - Incorrect: Translates a design into code. See PDF p. 18, 116.

Primary note proof

Study-note section SE21

System design turns the analysis model into architecture: design goals, subsystem responsibilities, dependencies, hardware mapping, data management, access control, control flow and startup/shutdown/error policies. Detail/object design refines APIs, parameter/return types, exceptions, component choices and algorithms. A subsystem is a manageable logical collection with services and an interface; it need not be a physical replaceable artifact.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 18 | lines 13-20

System and software design: The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
Software design involves identifying and describing the fundamental software system abstractions
and their relationships. The goal of the design phase is to transform the requirements specified in
the SRS document into a structure that is suitable for implementation in some programming language. In technical terms, during the design phase the software architecture is derived from the
SRS document. Two distinctly different approaches are available: the traditional design approach
and the object-oriented design approach.

Software Engineering module | PDF page 116 | lines 16-29

The analysis model describes the system completely from the actors' point of view and serves as the basis of communication between the client and the developers. The analysis model, however, does not contain information about the internal structure of the system, its hardware configuration,
or, more generally, how the system should be realized. System design is the first step in this direction. System design results in the following products:
• List of design goals, describing the qualities of the system that developers should optimize
• Software architecture, describing the subsystem decomposition in terms of subsystem responsibilities, dependencies among subsystems, subsystem mapping to hardware, and major policy decisions such as control flow, access control, and data storage.
The design goals are derived from the nonfunctional requirements. Design goals guide the decisions to be made by the developers especially when trade-offs are needed. The subsystem decomposition constitutes the bulk of system design. Developers divide the system into

manageable pieces to deal with complexity: Each subsystem is assigned to a team and realized independently. In order for this to be possible, though, developers need to address system-wide

Question sources

Computer Science Exit Exam practice collection · Question 7

Question 216 | CS0008

Instead of delivering the system as a single delivery, development and delivery is broken down into
phases with each phase delivering part of the required functionality. What type of development model is this?

A. Incremental

B. Waterfall

C. Spiral

D. Evolutionary

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer A: Incremental

Successive deliveries that each add part of the required usable functionality define incremental development. The product gains capability across these deliveries rather than waiting for a single completed-system handoff. Waterfall and a generic linear model organize phase sequence. Spiral is organized around risk assessment, and evolutionary development broadly describes refinement. Those can overlap with staged delivery, but incremental is the specific name for the delivery organization stated.

Why each choice is right or wrong

A - Correct: Incremental development adds usable required functionality in successive deliveries.

B - Incorrect: Waterfall names planned phase progression, not this particular staged-capability strategy.

C - Incorrect: Spiral names risk-driven cycles, even when increments are produced.

D - Incorrect: Evolutionary refinement is broader than the specific addition-of-functionality strategy described.

Primary note proof

Study-note section SE04

Choose incremental delivery when prioritized partial capability must reach users early. Smaller changes can be easier to debug and permit feedback, though architecture and integration still

require planning. Iteration need not rebuild the whole system every time, and increments need not be perfectly independent modules. Broad descriptions of early working software can fit prototypes, iterative or Agile processes; identify the purpose of that early version.

Study-note section SE06

Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 22 | lines 8-12

This model delivers software in small but usable pieces, called —increments. In general, each increment builds on those that have already been delivered. The model is iterative in nature and focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and
also provide a platform for evaluation by the user.

Question sources

Computer Science Exit Exam practice collection · Question 8

Question 217 | CS0009

We should NOT use the Waterfall software development process model in situations like:

A. Plenty of resources with required expertise are available freely

B. There are no ambiguous requirements

C. The project is short

D. Software requirements change quickly

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer D: Software requirements change quickly

Waterfall should be used only when requirements are well understood and unlikely to change radically. Rapid requirement change violates this selection condition.

Why each choice is right or wrong

A - Incorrect: Available expertise is not the documented contraindication. See PDF p. 20.

B - Incorrect: Unambiguous stable requirements support, rather than undermine, the model. See PDF p. 20.

C - Incorrect: The module says waterfall works well for smaller projects with well-understood requirements. See PDF p. 20.

D - Correct: Directly violates the stability assumption and creates costly rework. See PDF p. 20.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 20 | lines 18-22

> the development plan. Its major problem is the inflexible partitioning of the project into distinct stages. Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.
> In principle, the waterfall model should only be used when the requirements are well understood
> and unlikely to change radically during system development. However, the waterfall model

Question sources

Computer Science Exit Exam practice collection · Question 9

Question 218 | CS0010

Which one of the following is false about divide and conquer or modularization process?

A. Separate people can work on each sub-divided part

B. Opportunities arise for making the component reusable

C. Each individual component is smaller, and therefore easier to understand

D. Replacing or changing a component is difficult because it affects other parts extremely

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: Replacing or changing a component is difficult because it affects other parts extremely

Subsystems can be assigned to separate teams, reused and changed with reduced impact when coupled loosely. D contradicts the change-isolation goal of good modularization.

Why each choice is right or wrong

A - Incorrect: Independent teams can realize different manageable subsystems. See PDF p. 118, 116, 32, 120.

B - Incorrect: Reusable self-contained modules are an explicit OO benefit. See PDF p. 118, 116, 32, 120.

C - Incorrect: Decomposition produces simpler manageable units. See PDF p. 118, 116, 32, 120.

D - Correct: Strong propagation of every change describes undesirable tight coupling, not good modularization. See PDF p. 118, 116, 32, 120.

Primary note proof

Study-note section SE22

> Logical cohesion groups similar-category operations, often selected by a flag; temporal cohesion groups operations done at the same time; functional cohesion focuses on one task. Splitting an unrelated three-purpose component improves modularity. If behavior is preserved, the restructuring is refactoring; the words are not mutually exclusive answers. Smaller modules can aid understanding, independent work, reuse and maintenance, but too many interfaces also create complexity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 118 | lines 11-16

> 6.1.1. Subsystems and classes
> In order to reduce the complexity of the application domain, we identified smaller parts called classes and organized them into packages. Similarly, to reduce the complexity of the solution domain, we decompose a system into simpler parts, called subsystems, which are made of a number of solution domain classes. In the case of complex subsystems, we recursively apply this principle and decompose a subsystem into simpler subsystems (see Figure 6.2).

Software Engineering module | PDF page 116 | lines 16-29

> The analysis model describes the system completely from the actors' point of view and serves as the basis of communication between the client and the developers. The analysis model, however, does not contain information about the internal structure of the system, its hardware configuration,
> or, more generally, how the system should be realized. System design is the first step in this direction. System design results in the following products:
> • List of design goals, describing the qualities of the system that developers should optimize
> • Software architecture, describing the subsystem decomposition in terms of subsystem responsibilities, dependencies among subsystems, subsystem mapping to hardware, and major policy decisions such as control flow, access control, and data storage.
> The design goals are derived from the nonfunctional requirements. Design goals guide the decisions to be made by the developers especially when trade-offs are needed. The subsystem decomposition constitutes the bulk of system design. Developers divide the system into manageable pieces to deal with complexity: Each subsystem is assigned to a team and realized independently. In order for this to be possible, though, developers need to address system-wide

Software Engineering module | PDF page 32 | lines 12-21

A class in an object-oriented system carefully delineates between its interfaces the routines and attributes within a class are held together tightly. In a properly designed system, the classes will be grouped into subsystems but remain independent; therefore, changing one class has no impact
on other classes, and so, the impact is minimized. However, the object-oriented approach is not a panacea; nothing is magical here that will promote perfect design or perfect code.
Promotion of reusability
Objects are reusable because they are modeled directly out of a real-world problem domain. Each
object stands by itself or within a small circle of peers (other objects). Within this framework, the class does not concern itself with the rest of the system or how it is going to be used within a particular system.

Software Engineering module | PDF page 120 | lines 6-12

6.1.3. Coupling and coherence
Coupling is the strength of dependencies between two subsystems. If two subsystems are loosely coupled, they are relatively independent, and thus, modifications to one of the subsystems will have little impact on the other. If two subsystems are strongly coupled, modifications to one subsystem is likely to have impact on the other. A desirable property of a subsystem decomposition
is that subsystems are as loosely coupled as possible. This minimizes the impact that errors or future changes have on the correct operation of the system.

Question sources

Computer Science Exit Exam practice collection · Question 10

Question 219 | CS0013

For this project, planned testing effort in hours is setup + (cases × hours per case). The estimates (setup,cases,hours/case) are Unit=(4,80,0.1), Acceptance=(3,10,1), System=(6,20,0.5), Integration=(4,20,0.5). Which level has the greatest planned effort?

A. Unit testing

B. Acceptance testing

C. System testing

D. Integration testing

Hint

Use the effort formula given in the question separately for every level, then compare all computed totals.

Answer C: System testing

The estimates give unit effort 4+80×0.1=12 hours, acceptance 3+10×1=13, system 6+20×0.5=16, and integration 4+20×0.5=14. System testing therefore has the greatest planned effort for this project. The note does not make any test level universally longest; the explicit workload and setup assumptions are what justify this particular ranking.

Why each choice is right or wrong

A - Incorrect: Unit totals twelve hours, below the maximum.

B - Incorrect: Acceptance totals thirteen hours, below the maximum.

C - Correct: System totals sixteen hours, the largest estimate.

D - Incorrect: Integration totals fourteen hours, below system testing.

Primary note proof

Study-note section SE35

> Unit testing isolates a function, class or small component, often replacing dependencies with stubs/mocks; a driver invokes the unit and a stub substitutes a called dependency. Integration testing exercises interfaces and interactions among combined parts. System testing checks the complete integrated system against functional and nonfunctional requirements in a representative environment.

Study-note section SE35

> None of these levels is universally the longest.

Study-note section SE28

> Planning is continuous: identify scope/deliverables, decompose work, estimate effort/duration/cost, assign resources, sequence dependencies and revise as evidence changes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Software Engineering module | PDF page 148 | lines 19-26

> Unit testing focuses on the building blocks of the software system, that is, objects and subsystems.
> There are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall test activities, allowing us to focus on smaller units of the system. Second, unit testing makes it easier to pinpoint and correct faults, given that few components are involved in the test. Third, unit testing allows parallelism in the testing activities; that is, each component can be tested independently of one another.
> The specific candidates for unit testing are chosen from the object model and the system decomposition of the system. In principle, all the objects developed during the development

Software Engineering module | PDF page 144 | lines 17-32

> Figure 6.2 depicts an overview of the testing activities:
> • Unit testing tries to find faults in participating objects and/or subsystems with respect to

the use cases from the use case model
• Integration testing is the activity of finding faults when testing the individually tested components together, for example, subsystems described in the subsystem decomposition, while executing the use cases and scenarios from the RAD.
o The system structure testing is the culmination of integration testing involving all components of the system. Integration tests and structure tests exploit knowledge from the SDD using an integration strategy described in the Test Plan (TP).
• System testing tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
o Functional testing tests the requirements from the RAD and, if available, from the user manual.
o Performance testing checks the nonfunctional requirements and additional design goals from the SDD. Note that functional and performance testing are both done by

Question sources

Computer Science Exit Exam practice collection · Question 13

Question 220 | NS0293

A filled composition diamond is at Car's end of its Engine association. In this model Engine remains owned for its entire lifetime and cannot be detached or transferred. What does this imply?

A. Car uses Engine temporarily

B. Engine can exist independently of the Car

C. Engine is part of Car and cannot exist independently under the stated lifetime policy.

D. Car inherits from Engine

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Engine is part of Car and cannot exist independently under the stated lifetime policy.

Composition makes Engine a strongly owned part of Car, with the filled diamond beside the whole. Parts still owned are destroyed when their composite is destroyed. Since this model does not permit Engine to detach or transfer, it cannot continue an independent lifetime without Car. Temporary use would be a dependency, and inheritance would be shown by a hollow triangle.

Why each choice is right or wrong

A - Incorrect: Temporary use is a dependency rather than composition.

B - Incorrect: The no-detachment policy excludes an independent Engine lifetime.

C - Correct: Engine is Car's owned part and cannot outlive it under this policy.

D - Incorrect: Inheritance uses a hollow triangle.

Primary note proof

Study-note section SE18

> Composition uses a filled diamond at the whole end and strong ownership: a part is in at most one composite at a time, and parts still owned are destroyed with that composite. Parts may be detached/transferred under a defined model. Thus "cannot ever exist independently" is an oversimplification. For Car composing Engine, the diamond belongs beside Car; saying it points to Engine is imprecise. The Car-Engine example describes composition.

Question sources

Computer Science Exit Exam · 2018 · Question 10

Question 221 | NS0328

From the following statement which one is disadvantage of waterfall model?

A. model is simple and easy to understand and use

B. It is easy manage due to the rigidity of the model

C. Not suitable for projects where requirements are at a moderate to high risk of changing

D. In this model phases are processed and completed one at a time

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Poor suitability when requirements have substantial change risk.

SE03 says waterfall fits stable understood requirements and familiar technology. Late feedback can make changes costly, so moderate-to-high requirement-change risk is a direct disadvantage. Simplicity and milestone visibility are advantages. Sequential phase organization characterizes the classical model; it can contribute to rigidity, but C explicitly identifies the adverse scenario asked for.

Why each choice is right or wrong

A - Incorrect: Simplicity and ease of understanding are presented as advantages rather than the adverse scenario.

B - Incorrect: Manageable milestones can be an advantage of planned rigidity, despite its trade-offs.

C - Correct: Likely requirement changes make late feedback and rigid commitments costly.

D - Incorrect: Sequential phases describe the model; C more directly states the requested disadvantage.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Question sources

Computer Science Exit Exam · 2018 · Question 55

Question 222 | NS0333

For a search service whose functions and database technology are already specified, which requirement adds a quantitative service-speed target?

A. The system allow user to generate PDF report.

B. The system respond to user's request within 2 second under normal load

C. The system allow user to store data on relational database

D. The system allow user to submit emergency report

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: The system respond to user's request within 2 second under normal load

The two-second response obligation bounds the speed of a service under the stated load, so it is a performance requirement. Generating a PDF and submitting an emergency report describe services the system provides. The database clause describes storage behavior together with an implementation choice; it does not impose a response-time threshold. The explicit performance scope removes the broader taxonomy overlap between nonfunctional requirements and technology constraints.

Why each choice is right or wrong

A - Incorrect: Report generation specifies a service, not its response-time limit.

B - Correct: A two-second response bound under normal load is a quantitative speed target.

C - Incorrect: This specifies storage behavior and a database technology, without a timing bound.

D - Incorrect: Submitting an emergency report specifies a function rather than its speed.

Primary note proof

Study-note section SE10

> Nonfunctional requirements include quality attributes and constraints. Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning. Security concerns protection; compatibility concerns cooperating environments; portability concerns moving environments; maintainability/supportability concern change, diagnosis and evolution.

Study-note section SE10

> "Search responds within two seconds under 100 concurrent users" is measurable performance. "New users finish checkout within one minute without help" is usability. "Use Java" is a technology/pseudo-requirement, often separately classified as a system constraint. Dependability includes reliability, safety and robustness; portability is not its usual central category. Performance degradation at peak load primarily concerns performance, not automatically reliability.

Question sources

Computer Science Exit Exam · 2018 · Question 61

Question 223 | NS0347

Which one of the following UML building block that defines the static part of the model and represents physical and conceptual elements?

A. Structural things

B. Behavioral things

C. An notational things

D. Grouping things

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer A: Structural things.

SE17 explicitly describes structural things as conceptual or physical structural elements. They make up the static model in contrast with behavior over time. Behavioral things cover dynamic aspects, grouping organizes elements and annotations add explanatory information rather than being the static elements themselves. Thus A directly matches both parts of the definition.

Why each choice is right or wrong

A - Correct: Structural things are the static conceptual or physical building blocks described.

B - Incorrect: Behavioral things concern dynamic behavior rather than the static structural elements.

C - Incorrect: Annotations add explanatory information rather than defining the static structural building blocks.

D - Incorrect: Grouping organizes model elements rather than being the stated conceptual/physical structural category.

Primary note proof

Study-note section SE17

> A class box lists name, attributes and operations; + public, - private, # protected and ~ package indicate visibility. A class diagram shows static types and relationships, while an object diagram shows particular instances/links. UML is maintained and standardized by OMG; it is a modeling language, not a lifecycle process. Structural things describe conceptual or physical structural elements.

Study-note section SE-S03

> In the course's UML building-block terminology, structural things describe static conceptual or physical elements. Behavioral things describe dynamic interactions and state behavior.

Grouping things organize model elements into packages. Annotational things attach explanatory remarks or constraints, usually as notes.

Question sources

Computer Science Exit Exam · 2018 · Question 80

Question 224 | NS0740

What is the difference between functional and non-functional requirements? Give an example of each.

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer: Functional requirements specify observable behavior; nonfunctional requirements specify qualities or constraints. Example: generate a PDF report versus respond within two seconds under a stated load.

SE09 includes required and prohibited services, so generating a PDF report is functional. SE10 specifies qualities such as performance, usability and availability; a response-time threshold under defined load is nonfunctional. Security mechanisms can implement functional behavior as well as support quality goals, so a negative or security-related statement is not automatically nonfunctional.

Primary note proof

Study-note section SE09

> A functional requirement states an observable service or behavior, including reactions to inputs, errors and forbidden actions. “When the sensor detects intrusion, send an alert” is functional.

Study-note section SE09

> “The system shall generate a PDF report” is functional. “The system shall not reveal another customer’s balance” can be a security-related functional rule. Security as a broad quality goal and the functions implementing it must be distinguished.

Study-note section SE10

> Nonfunctional requirements include quality attributes and constraints. Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning.

Study-note section SE10

> “Search responds within two seconds under 100 concurrent users” is measurable performance. “New users finish checkout within one minute without help” is usability.

Question sources

Software Engineering review questions · Question 1

Question 225 | NS0741

Why is low coupling considered good in software design?

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer: Low coupling limits dependencies, so changes remain local and modules are easier to understand, replace, test and maintain.

SE22 measures coupling through dependencies between modules and recommends low coupling with high cohesion. Shared mutable data and knowledge of another module's representation increase the chance that one change affects others. Stable interfaces and information hiding constrain that impact. More interfaces can add complexity, so minimizing harmful dependencies matters more than simply maximizing the number of modules.

Primary note proof

Study-note section SE22

> Cohesion measures how well elements inside a module serve a related purpose; coupling measures dependency between modules. Seek high cohesion and low coupling. Shared mutable data and knowledge of another module's internal representation increase coupling. Information hiding exposes stable operations while concealing changeable decisions.

Question sources

Software Engineering review questions · Question 2

Question 226 | NS0742

What is the difference between Waterfall and Agile methodologies?

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer: Waterfall uses planned, reviewed stages and suits stable requirements; Agile emphasizes collaboration, working results, feedback and adaptation.

SE03 describes planned phases with costly late feedback, useful for stable requirements and familiar technology. SE05 values customer participation and adaptation with useful working increments. Agile still plans and documents; waterfall can revisit work under control. The practical difference is feedback and commitment organization, not a claim that one process forbids change or the other forbids documentation.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Study-note section SE05

> Agile favors people collaborating, useful working results, customer participation and adaptation over rigid adherence to procedures, exhaustive documents, contracts or fixed plans. The latter still have value. Agile does not mean no planning or no documentation. A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Question sources

Software Engineering review questions · Question 3

Question 227 | NS0743

What is the difference between <<include>> and <<extend>> in use case diagrams?

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer: Include factors required reused behavior, with base→included; extend adds conditional behavior, with extension→base.

SE16 uses include for reused required behavior incorporated when the inclusion point is reached. Extend adds behavior at extension points under a condition, and the base remains meaningful independently. The dashed arrows have different directions: including use case toward included, and extending use case toward base. Neither relation should be confused with actor association or specialization generalization.

Primary note proof

Study-note section SE16

> Include factors required reused behavior: Submit Order includes Validate Order, with the dashed arrow from the including/base use case to the included one. Extend adds conditional behavior at extension points: Apply Discount extends Checkout, with the arrow from the extension to the base. The base is meaningful independently. For an extension, draw the arrow from the extending use case to the base use case.

Study-note section SE16

> Generalization relates a specialized actor/use case to a more general one, with a hollow triangle toward the general element. An abstract use case specifies behavior for specialization and is not directly instantiated as a complete use case. It is not defined merely by reuse or lack of a direct actor line. Included behavior is incorporated where the inclusion is reached; do not infer every conditional branch in the base always executes.

Question sources

Software Engineering review questions · Question 4

Question 228 | NS0744

Why is maintenance often the most expensive phase of the SDLC?

Hint

Classify the motive for the change before its implementation: repair, environment, useful capability or reduced future risk.

Answer: Maintenance can dominate cost because it lasts years and repeatedly requires legacy understanding, changes, regression testing and release work.

SE40 says maintenance often spans years and includes understanding existing code plus regression effort. Work is not limited to bug fixes: environments, useful capabilities and future risks also change. Repeated impact analysis, implementation and testing accumulate across the operational lifetime, so maintenance may dominate lifecycle cost. This is a common outcome, not a guarantee for every project.

Primary note proof

Study-note section SE40

> Corrective maintenance fixes faults; adaptive maintenance responds to environment changes; perfective maintenance improves useful capability/performance; preventive maintenance reduces future risk, such as restructuring fragile code. Motive determines category; refactoring can support preventive or perfective work. Maintenance often lasts years and includes understanding legacy code and regression effort, so it may dominate lifecycle cost.

Study-note section SE40

> A typical evolution sequence is change request→impact analysis→release planning→implementation/testing→release. Among a list lacking the request step, impact analysis may be the earliest listed technical activity. Major restructuring is reengineering; if the architecture itself changes, architectural transformation is the more specific description. Markets, technology, stakeholders and funding can all change scope or priorities.

Question sources

Software Engineering review questions · Question 5

Question 229 | NS0745

Which software development approach emphasizes breaking systems into functions or procedures that manipulate isolated data?

A. Object-Oriented Approach

B. Traditional Approach

C. Agile Approach

D. Iterative Approach

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Traditional/procedural decomposition.

SE01 contrasts procedural decomposition organized around functions with object-oriented decomposition organized around objects combining state and behavior. B is the available traditional/procedural label for functions operating on data. Agile and iterative describe lifecycle organization rather than this decomposition paradigm, and either can be used with procedural or object-oriented programming.

Why each choice is right or wrong

A - Incorrect: Object orientation combines state and behavior in objects rather than organizing primarily around separate procedures.

B - Correct: Traditional/procedural decomposition organizes functions and procedures acting on data.

C - Incorrect: Agile concerns process adaptation rather than requiring function-based decomposition.

D - Incorrect: Iteration describes revisiting development work and does not determine procedural versus object decomposition.

Primary note proof

Study-note section SE01

> Procedural decomposition organizes functions; object oriented decomposition organizes objects with state and behavior. Either can use iterative or sequential development. OOP does not automatically shorten every project. Reuse, encapsulation and consistent models can reduce duplication and ease change; poor object design can still be complex. Functional programming emphasizes function evaluation and controlled side effects; event-driven programs react to events.

Question sources

Software Engineering review questions · Question 1

Question 230 | NS0746

In the Object-Oriented Approach, which property allows classes to be used in different parts of the system or other projects without rewriting them?

A. Encapsulation

B. Polymorphism

C. Reusability

D. Abstraction

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Reusability.

The stem explicitly asks about using existing classes elsewhere without rewriting them, which is reuse. SE01 lists reuse alongside encapsulation and consistent models as ways to reduce duplication and ease change. Encapsulation and abstraction can make reuse safer, while polymorphism supports compatible varying implementations, but those are enabling concepts rather than the name of the capability described.

Why each choice is right or wrong

A - Incorrect: Encapsulation hides implementation and can enable reuse, but it is not the capability's name.

B - Incorrect: Polymorphism supports differing implementations through compatible interfaces rather than directly naming reuse.

C - Correct: Reusability means using suitable existing classes or components in other contexts without rewriting them.

D - Incorrect: Abstraction exposes relevant contracts and can assist reuse, but the requested property is reusability.

Primary note proof

Study-note section SE-S02

> Encapsulation groups state with operations and hides representation behind an interface. Polymorphism permits different object types to respond differently through a common operation or interface. Abstraction exposes the relevant contract while omitting unnecessary detail. Reusability means using suitable existing classes or components in new contexts rather than writing them again. These ideas support each other but name different properties.

Question sources

Software Engineering review questions · Question 2

Question 231 | NS0747

Which of the four fundamental software process activities involves making updates to meet changing customer needs after deployment?

A. Software Specification

B. Software Design and Implementation

C. Software Validation

D. Software Evolution

Hint

Classify the motive for the change before its implementation: repair, environment, useful capability or reduced future risk.

Answer D: Software evolution.

SE40 describes maintenance and evolution through change requests, impact analysis, planning, implementation/testing and release. Updating a deployed product to meet changing customer needs belongs to evolution among the four broad process activities. Specification, implementation and validation still occur within that work, but D names the lifecycle-wide change activity asked for rather than one constituent task.

Why each choice is right or wrong

A - Incorrect: Specification captures needs but does not name the overall postdeployment change activity.

B - Incorrect: Design/implementation is part of making a change rather than the broad evolution activity.

C - Incorrect: Validation checks the changed product against needs rather than naming ongoing adaptation itself.

D - Correct: Evolution updates existing software as stakeholder needs and operating conditions change.

Primary note proof

Study-note section SE40

> Corrective maintenance fixes faults; adaptive maintenance responds to environment changes; perfective maintenance improves useful capability/performance; preventive maintenance reduces future risk, such as restructuring fragile code. Motive determines category; refactoring can support preventive or perfective work. Maintenance often lasts years and includes understanding legacy code and regression effort, so it may dominate lifecycle cost.

Study-note section SE40

> A typical evolution sequence is change request→impact analysis→release planning→implementation/testing→release.

Question sources

Software Engineering review questions · Question 3

Question 232 | NS0748

A software process that plans all activities in advance and measures progress against that plan is classified as:

A. Agile Process

B. Plan-Driven Process

C. Incremental Process

D. Evolutionary Process

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Plan-driven process.

The stem defines up-front planning and measurement against that plan. B names the plan-driven organization, illustrated by SE03's planned phases and reviewed deliverables. Incremental delivery and evolution describe how results grow or change and can coexist with planning. Agile also plans, but it emphasizes adaptation rather than defining all activity commitments in advance as stated here.

Why each choice is right or wrong

A - Incorrect: Agile uses planning but adapts content as feedback arrives rather than defining this fixed up-front emphasis.

B - Correct: Plan-driven explicitly organizes activities and progress measurement around an advance plan.

C - Incorrect: Incremental describes staged capability delivery and does not alone specify up-front planning commitments.

D - Incorrect: Evolutionary describes progressive change rather than necessarily advance planning of all activities.

Primary note proof

Study-note section SE-S04

> In a plan-driven process, activities are planned in advance and progress is assessed against that plan. Agile planning is incremental and adapts to changing needs. This distinction concerns planning strategy; a process can combine advance planning with staged deliveries and feedback.

Question sources

Software Engineering review questions · Question 4

Question 233 | NS0749

In requirement elicitation, the document created to describe what the system should do - agreed upon by clients and developers - is called:

A. Analysis Model

B. Use Case Diagram

C. Requirements Specification

D. Design Document

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Requirements Specification.

SE08 states that the Software Requirements Specification organizes agreed requirements. That is the artifact describing what the system should do for client/developer agreement. Analysis models support technical discussion, a use-case diagram captures only selected goals/relationships and a design document describes solution decisions. None replaces the overall agreed requirements specification.

Why each choice is right or wrong

A - Incorrect: Analysis models formalize selected aspects for discussion rather than replacing the full agreed requirements document.

B - Incorrect: A use-case diagram shows goals and roles but not the complete requirements specification.

C - Correct: The requirements specification organizes the agreed services and constraints the system must satisfy.

D - Incorrect: A design document describes how the solution is organized rather than the complete statement of required behavior.

Primary note proof

Study-note section SE08

> Business requirements explain why the project exists and its goals. User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints. The Software Requirements Specification (SRS) organizes the agreed requirements. A natural-language specification communicates with clients while analysis models formalize it for technical discussion; real specifications can also contain diagrams.

Question sources

Software Engineering review questions · Question 5

Question 234 | NS0750

What is the key distinction between a Requirements Specification and an Analysis Model?

A. The specification uses diagrams; the model uses natural language

B. The specification uses natural language; the model uses symbols and diagrams

C. Both use the same representation but serve different purposes

D. The analysis model is only for developers; specification is for testing

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: In the taught convention, natural-language client specification versus symbolic/diagrammatic analysis models.

SE08 explicitly describes natural-language specification for clients and analysis models that formalize it for technical discussion. B matches that teaching convention. This is not a universal ban on diagrams in an SRS: the note immediately allows them in real specifications. The other choices reverse the convention or impose unsupported exclusive audiences and purposes.

Why each choice is right or wrong

A - Incorrect: It reverses the natural-language specification versus formalized-model convention taught in the note.

B - Correct: This matches the taught convention, while real requirements specifications can also include diagrams.

C - Incorrect: The taught distinction does use different representation emphases rather than identical forms.

D - Incorrect: Neither artifact has the exclusive audience or testing-only purpose asserted here.

Primary note proof

Study-note section SE08

> Business requirements explain why the project exists and its goals. User requirements describe stakeholder needs at a useful high level; system requirements specify detailed services and constraints. The Software Requirements Specification (SRS) organizes the agreed requirements. A natural-language specification communicates with clients while analysis models formalize it for technical discussion; real specifications can also contain diagrams.

Question sources

Software Engineering review questions · Question 6

Question 235 | NS0751

In use case modeling, actors can be defined as:

A. Only human users who interact with the system

B. People, machines, or external systems that interact with the system

C. Only primary users who need system assistance

D. Software components within the system boundary

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: External interacting roles played by people, devices or other systems.

SE14 explicitly permits a person, device or other system to play an external actor role. B is the only option including those categories. Actors are not restricted to primary users or humans, and internal components are inside the chosen system boundary rather than external roles. One person may play several actors, so actor instances should not be confused with individual people.

Why each choice is right or wrong

A - Incorrect: Actors can be devices or external systems as well as human users.

B - Correct: People, devices and external systems can play roles interacting across the system boundary.

C - Incorrect: Supporting and administrative external roles can be actors, not only primary goal-seeking users.

D - Incorrect: Internal components are not external actor roles under the chosen system boundary.

Primary note proof

Study-note section SE14

> Draw actors outside the system boundary rectangle and use-case ovals inside. Name a goal with a verb and noun. The diagram shows roles, goals and relationships, not the full detailed algorithm or UI layout. Being an internal component is different from being an external actor; the chosen boundary determines which role applies.

Study-note section SE14

> An actor is an external role interacting with the system; it can be a person, device or another system.

Question sources

Software Engineering review questions · Question 7

Question 236 | NS0752

Which type of requirement specifies that 'the system must respond within 1 second'?

A. Functional Requirement

B. System Constraint

C. Non-Functional Requirement

D. Domain Requirement

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Nonfunctional performance requirement.

SE10 identifies response time as a performance quality. A one-second threshold constrains how well a service responds rather than naming the service itself, so C is the appropriate category. A complete measurable requirement should also define the operation, load and timing conditions. Those omissions affect test precision, not the underlying classification of the stated response-time target.

Why each choice is right or wrong

A - Incorrect: The statement constrains response speed rather than specifying which service is performed.

B - Incorrect: The specific quality is performance; this is not primarily a technology or implementation constraint.

C - Correct: The one-second limit is a nonfunctional response-time/performance target.

D - Incorrect: No domain-specific business rule is stated; the obligation concerns a general performance quality.

Primary note proof

Study-note section SE10

> Nonfunctional requirements include quality attributes and constraints. Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning. Security concerns protection; compatibility concerns cooperating environments; portability concerns moving environments; maintainability/supportability concern change, diagnosis and evolution.

Question sources

Software Engineering review questions · Question 8

Question 237 | NS0753

When technology constraints are separated from quality attributes, what is the most specific category for “the system must be built using Java”?

A. Functional Requirement

B. Non-Functional Requirement

C. System/Constraint Requirement

D. Quality Requirement

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: System/Constraint Requirement

The statement prescribes an implementation technology rather than an externally observable service or a measurable product quality. In the course’s explicit separation of quality attributes from technology constraints, the specific category is a system or constraint requirement. A broad requirements taxonomy can place constraints under nonfunctional requirements, but the question asks for the more specific category. No performance, reliability or usability threshold is supplied by merely naming Java.

Why each choice is right or wrong

A - Incorrect: The statement prescribes technology rather than an observable service.

B - Incorrect: This is a broader umbrella in some taxonomies; it is incomplete when the specific constraint category is offered.

C - Correct: A required implementation language is a technology or system constraint.

D - Incorrect: No measurable quality attribute is specified merely by requiring Java.

Primary note proof

Study-note section SE10

> “Search responds within two seconds under 100 concurrent users” is measurable performance. “New users finish checkout within one minute without help” is usability. “Use Java” is a technology/pseudo-requirement, often separately classified as a system constraint. Dependability includes reliability, safety and robustness; portability is not its usual central category. Performance degradation at peak load primarily concerns performance, not automatically reliability.

Question sources

Software Engineering review questions · Question 9

Question 238 | NS0754

Which nonfunctional requirement category addresses the ability of the system to work across different web browsers and devices?

A. Reliability

B. Scalability

C. Usability

D. Compatibility

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer D: Compatibility.

SE10 distinguishes compatibility, which concerns cooperating environments, from portability, which concerns moving environments. Supporting different browsers and devices in the question is the offered compatibility category. Reliability concerns failure-free service, scalability concerns growth and usability concerns task success/ease of learning. None of those names the stated cross-environment cooperation requirement as directly.

Why each choice is right or wrong

A - Incorrect: Reliability concerns failure-free service rather than which browsers and devices can cooperate with the system.

B - Incorrect: Scalability concerns handling increased workload rather than cross-browser/device compatibility.

C - Incorrect: Usability concerns ease of learning and successful tasks, not the supported environment set itself.

D - Correct: Compatibility addresses operating/cooperating across the specified browsers and device environments.

Primary note proof

Study-note section SE10

> Nonfunctional requirements include quality attributes and constraints. Performance concerns response time, throughput and resource use; reliability concerns failure-free service; availability concerns being ready for use; usability concerns task success and ease of learning. Security concerns protection; compatibility concerns cooperating environments; portability concerns moving environments; maintainability/supportability concern change, diagnosis and evolution.

Study-note section SE-S07

> Scalability is the ability to accommodate additional workload while maintaining the required service, often by increasing available resources. It concerns growth in demand. Compatibility concerns supported cooperating environments, such as browsers and devices; usability concerns successful and learnable user tasks.

Question sources

Software Engineering review questions · Question 10

Question 239 | NS0755

Dependability in software engineering is a broader concept that includes all of the following EXCEPT:

A. Reliability

B. Robustness

C. Safety

D. Portability

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer D: Portability is the exception to the usual central dependability categories.

SE10 explicitly includes reliability, safety and robustness in dependability and says portability is not its usual central category. Portability instead concerns moving environments. Thus D is the listed exception. The distinction does not imply portability never affects practical service outcomes, only that it is a different quality category from the taught dependability grouping.

Why each choice is right or wrong

A - Incorrect: Reliability is explicitly included in the taught dependability grouping.

B - Incorrect: Robustness is explicitly included in the broader dependability concept.

C - Incorrect: Safety is explicitly included among dependability concerns.

D - Correct: Portability concerns moving environments rather than the usual central dependability categories.

Primary note proof

Study-note section SE10

> Dependability includes reliability, safety and robustness; portability is not its usual central category.

Study-note section SE10

> Security concerns protection; compatibility concerns cooperating environments; portability concerns moving environments; maintainability/supportability concern change, diagnosis and evolution.

Study-note section SE-S07

> Robustness concerns reliable operation across varied or difficult operating conditions and controlled failure outside the supported range.

Question sources

Software Engineering review questions · Question 11

Question 240 | NS0756

In requirements elicitation activities, which step involves formalizing scenarios into structured descriptions of system interactions?

A. Identifying Actors

B. Identifying Use Cases

C. Identifying Scenarios

D. Refining Use Cases

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: Identifying use cases by abstracting concrete scenarios.

SE13 calls a scenario one concrete execution and a use case an abstraction of related success and alternative scenarios. Formalizing those interactions into the more general use-case description therefore identifies use cases. Identifying actors establishes external roles, identifying scenarios captures individual executions and later refinement adds detail to an already identified use case.

Why each choice is right or wrong

A - Incorrect: Actor identification establishes external roles rather than generalizing scenario executions into interaction descriptions.

B - Correct: Use cases abstract related concrete scenarios into structured goal-oriented interaction descriptions.

C - Incorrect: Scenario identification captures concrete executions rather than their use-case abstraction.

D - Incorrect: Refinement elaborates an existing use case; the scenario-to-use-case abstraction is the identification step asked here.

Primary note proof

Study-note section SE13

> A scenario is one concrete execution; a use case abstracts the related successful and alternative scenarios. Analysis integrates functional (use cases/scenarios), structural/object (classes) and dynamic (states/interactions) models. Entity objects represent domain information such as Incident; boundary objects mediate interaction, such as ReportForm; control objects coordinate a use case. A UI form is normally boundary, not entity.

Question sources

Software Engineering review questions · Question 12

Question 241 | NS0757

According to the Simple Use Case Writing Guide, which of the following is CORRECT practice?

A. Use passive voice so responsibility is unclear

B. Describe user interface elements in detail

C. Document exceptions separately from the main flow

D. Combine multiple tasks into one use case for brevity

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Document exceptions separately from the main success flow.

SE15 instructs writers to use active subject-verb steps and separate errors from normal flow. C follows that rule directly. Passive voice hides responsibility, detailed UI design distracts from goal-level behavior, and combining unrelated tasks does not produce a clear use case. The distinction concerns organizing scenarios rather than omitting exception handling.

Why each choice is right or wrong

A - Incorrect: Passive voice makes responsibility less clear, contrary to the active subject-verb guideline.

B - Incorrect: Detailed UI elements are not the goal-level behavior and flow structure emphasized by the guide.

C - Correct: Separating exception behavior keeps the main success flow clear while still documenting failures.

D - Incorrect: Combining multiple unrelated tasks sacrifices a coherent goal and consistent abstraction level.

Primary note proof

Study-note section SE15

> Example: Submit Incident. Precondition: officer authenticated. 1 Officer supplies location and description. 2 System validates details. 3 System records the incident. 4 System returns its ID. Alternative 2a: location missing → report the problem and request correction; save nothing. Success postcondition: incident and ID persist; failure guarantee: no partial record. An alternate flow is a conditional variation; an exception flow handles a failure. Preconditions hold before entry, while a trigger starts the use case.

Study-note section SE15

> Use active subject-verb steps at one abstraction level; separate errors from the normal flow.

Question sources

Software Engineering review questions · Question 13

Question 242 | NS0758

In the FAST (Facilitated Application Specification Technique), participants are initially asked to individually create lists of three item categories. Which is NOT one of them?

A. Things that are part of the environment surrounding the system

B. Things the system produces or outputs

C. Technical risks and constraints

D. Things the system uses or takes as inputs

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Technical risks and constraints are not one of the three initial FAST lists.

SE12 explicitly lists environment elements, outputs and inputs as the initial participant categories. A, B and D match them. Technical risks and constraints can matter elsewhere in engineering, but they are not one of those three named initial FAST lists. The negative question asks about that procedure, not whether risk information is ever useful.

Why each choice is right or wrong

A - Incorrect: Environment elements are explicitly one of the initial FAST categories.

B - Incorrect: System outputs are explicitly included in the three initial lists.

C - Correct: Risks and constraints are not one of the specified environment/output/input categories.

D - Incorrect: Inputs and things the system uses are part of the stated initial categories.

Primary note proof

Study-note section SE12

> In the taught FAST technique, participants list environment elements, outputs and inputs, consolidate lists and develop smaller specifications. QFD prioritizes customer value: normal needs are explicitly stated, expected needs are assumed, and exciting needs provide unanticipated benefit. Brainstorm first to generate ideas, then evaluate and prioritize; do not confuse that with letting every idea become a committed requirement.

Question sources

Software Engineering review questions · Question 14

Question 243 | NS0759

In Quality Function Deployment (QFD), which category of requirements represents features that go beyond customer expectations and provide great satisfaction?

A. Normal Requirements

B. Expected Requirements

C. Exciting Requirements

D. Implicit Requirements

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Exciting requirements.

SE12 distinguishes explicitly stated normal needs, assumed expected needs and exciting needs that bring unanticipated benefit. Features exceeding expectations and producing additional satisfaction match the exciting category. Calling them merely implicit would confuse customer assumptions with unexpected added value rather than identify the QFD term.

Why each choice is right or wrong

A - Incorrect: Normal requirements are explicitly stated needs rather than unexpected extra benefits.

B - Incorrect: Expected requirements are assumed by customers and do not denote surprising added satisfaction.

C - Correct: Exciting requirements deliver unanticipated benefit beyond the customer's expectations.

D - Incorrect: Implicit describes assumed needs more generally, not the QFD category for unanticipated benefits.

Primary note proof

Study-note section SE12

> In the taught FAST technique, participants list environment elements, outputs and inputs, consolidate lists and develop smaller specifications. QFD prioritizes customer value: normal needs are explicitly stated, expected needs are assumed, and exciting needs provide unanticipated benefit. Brainstorm first to generate ideas, then evaluate and prioritize; do not confuse that with letting every idea become a committed requirement.

Question sources

Software Engineering review questions · Question 15

Question 244 | NS0760

Which validation property ensures that the requirements specification does not contain contradictory statements?

A. Completeness

B. Consistency

C. Clarity

D. Correctness

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: Consistency.

SE11 directly defines consistency as excluding contradictions. Completeness covers agreed scope, clarity gives one intended interpretation, and correctness reflects stakeholder needs. A specification can be complete yet inconsistent, or clear yet wrong, so these properties cannot substitute for the precise no-contradictions condition.

Why each choice is right or wrong

A - Incorrect: Completeness concerns covering scope and behavior, not whether requirements contradict each other.

B - Correct: Consistency requires the specification’s statements to coexist without contradiction.

C - Incorrect: Clarity addresses interpretive ambiguity rather than logical compatibility between requirements.

D - Incorrect: Correctness concerns actual stakeholder needs, a different validation property.

Primary note proof

Study-note section SE11

> Correct requirements reflect actual stakeholder needs. Complete requirements address the agreed scope, normal and exceptional behavior. Consistency excludes contradictions. Clarity/unambiguity gives one intended interpretation; feasibility/realism means achievable within constraints. Verifiability means objective evidence can check fulfillment. Replace “fast for most users” with a defined metric, workload and threshold.

Question sources

Software Engineering review questions · Question 16

Question 245 | NS0761

The Waterfall model is MOST appropriate when:

A. Requirements are expected to change frequently during development

B. The project is large and complex with evolving needs

C. Requirements are constant, the project is short, and technology is stable

D. The team needs to deliver working software in small increments

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Stable requirements and familiar, stable technology.

SE03 says classical waterfall fits well-understood stable requirements and familiar technology, directly supporting C among these alternatives. Frequent change, evolving needs and early small deliveries call for feedback or incremental capability rather than waterfall's late feedback. Short duration alone is not a proof, but C also supplies the decisive stability conditions stated in the note.

Why each choice is right or wrong

A - Incorrect: Frequent requirement changes make late-feedback sequential planning a poor fit.

B - Incorrect: Large evolving needs do not establish the stable requirements that favor classical waterfall.

C - Correct: Constant requirements and stable technology match the note's stated waterfall selection conditions.

D - Incorrect: Prioritized usable small deliveries are the rationale for incremental delivery rather than a single sequential release.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Question sources

Software Engineering review questions · Question 17

Question 246 | NS0762

Which is a characteristic drawback of classical waterfall compared with approaches that seek repeated early feedback?

A. Requires skilled developers

B. Late feedback can make requirement changes costly through rework of earlier phase deliverables.

C. No clear project milestones

D. Does not support documentation

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Late feedback can make requirement changes costly through rework of earlier phase deliverables.

Classical waterfall separates planned phases and may receive important feedback late. Correcting earlier assumptions can then require costly rework across completed deliverables, which is its characteristic drawback here. Changes are still possible under control. Skilled developers are needed across approaches, and waterfall provides milestone visibility and documentation, so the other alternatives do not identify the stated comparative disadvantage.

Why each choice is right or wrong

A - Incorrect: Skill requirements are not unique to waterfall.

B - Correct: Late feedback can make requirement changes costly after earlier phases.

C - Incorrect: Waterfall provides clear reviewed milestones.

D - Incorrect: Waterfall supports planned documentation.

Primary note proof

Study-note section SE03

> Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly. It does not make change physically impossible: real projects revisit work under control, and modified waterfall explicitly includes feedback. Rigidity offers milestone visibility but can freeze incorrect assumptions.

Question sources

Software Engineering review questions · Question 18

Question 247 | NS0763

What is the key conceptual distinction between iteration and incremental delivery, even when a process combines both?

A. Iterative adds new features each cycle; Incremental refines the same set repeatedly

B. Iteration revisits and improves a solution; an increment adds usable functionality.

C. They are identical; the terms are interchangeable

D. Incremental involves risk analysis; Iterative does not

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Iteration revisits and improves a solution; an increment adds usable functionality.

Iteration revisits and improves a solution, while an increment adds usable functionality. A process can refine existing capability and add new capability in the same cycle, so the concepts are compatible but not identical. Neither requires rebuilding the entire system or making every increment completely independent. Risk analysis is a separate concern rather than the structural distinction defining these two terms.

Why each choice is right or wrong

A - Incorrect: This reverses refinement and added functionality.

B - Correct: Iteration refines; increments add usable capability.

C - Incorrect: The terms differ even when combined in one process.

D - Incorrect: Risk analysis does not define their distinction.

Primary note proof

Study-note section SE04

> Choose incremental delivery when prioritized partial capability must reach users early. Smaller changes can be easier to debug and permit feedback, though architecture and integration still require planning. Iteration need not rebuild the whole system every time, and increments need not be perfectly independent modules. Broad descriptions of early working software can fit prototypes, iterative or Agile processes; identify the purpose of that early version.

Study-note section SE04

> Iteration revisits and improves a solution; an increment adds usable functionality. A process may do both.

Question sources

Software Engineering review questions · Question 19

Question 248 | NS0764

A large application already has its core capabilities. The main development goal is to revisit and improve that existing solution repeatedly as feedback and future changes arrive, rather than primarily add new capability in each delivery. Which concept most directly names that emphasis?

A. Waterfall Model

B. V-Model

C. Iterative Model

D. Incremental Model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Iterative Model

The stated emphasis is iterative: revisiting and improving an existing solution in response to feedback. Incremental development names the addition of usable functionality, which is not the primary goal specified, although both can coexist. Classical waterfall's stable staged planning and the V-model's paired assurance structure do not directly name this refinement emphasis. Large size and anticipated change alone would not have selected a unique model.

Why each choice is right or wrong

A - Incorrect: Classical waterfall does not directly name repeated refinement as the organizing emphasis.

B - Incorrect: The V-model emphasizes paired development and tests.

C - Correct: Iteration directly names revisiting and improving the existing solution.

D - Incorrect: Incremental describes added functionality rather than the specified primary emphasis.

Primary note proof

Study-note section SE04

> Choose incremental delivery when prioritized partial capability must reach users early. Smaller changes can be easier to debug and permit feedback, though architecture and integration still require planning. Iteration need not rebuild the whole system every time, and increments need not be perfectly independent modules. Broad descriptions of early working software can fit prototypes, iterative or Agile processes; identify the purpose of that early version.

Study-note section SE02

> A common lifecycle names requirements, design, implementation, testing, deployment and maintenance. Analysis clarifies the problem; system design chooses architecture; object/detail design specifies interfaces and algorithms; implementation writes code; testing evaluates behavior; deployment puts it into service. These activities may overlap and repeat.

Study-note section SE04

> Iteration revisits and improves a solution; an increment adds usable functionality. A process may do both.

Question sources

Software Engineering review questions · Question 20

Question 249 | NS0765

In the Agile SDLC, small units of functionality that the software needs to deliver are called:

A. Sprints

B. Iterations

C. User Stories

D. Epics

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: User stories.

SE05 defines a user story as a small user need and value supported by acceptance criteria. Sprints and iterations name cycles or timeboxes, not the need itself. An epic groups broader work. C therefore names the small functional work unit intended here, without claiming that Agile requirements consist only of story text.

Why each choice is right or wrong

A - Incorrect: A Sprint is a timeboxed development interval rather than the individual user need.

B - Incorrect: An iteration is a cycle of work or refinement, not a small functional requirement.

C - Correct: A user story states a small user need and its value with acceptance criteria.

D - Incorrect: An epic groups broader work rather than naming the smaller story-sized need.

Primary note proof

Study-note section SE05

> Scrum uses timeboxed Sprints of at most one month, with a Sprint Goal, selected work and a usable Increment meeting a Definition of Done. Scope is discussed before and during planning; it is not forbidden to agree scope. Timeboxes tend to stay fixed while content is adapted without endangering the goal. Agile can be combined with required assurance/documentation; safety-critical context alone cannot prove it is unsuitable.

Study-note section SE05

> A user story states a small user need and value, supported by acceptance criteria; an epic groups broader work.

Question sources

Software Engineering review questions · Question 21

Question 250 | NS0766

The Spiral Model distinguishes itself from other models primarily through its emphasis on:

A. Sequential phase completion

B. Risk Analysis at every cycle

C. Customer delivery at the end of the project

D. Strict adherence to initial requirements

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer B: Risk analysis and reduction in every spiral cycle.

SE06 defines spiral development as risk driven, with each cycle identifying objectives and alternatives, assessing and reducing risks, producing a result and planning commitment. That emphasis distinguishes it from merely repeating phases or delivering only at the end. It can adapt requirements and combine other activities rather than enforcing an unchangeable initial specification.

Why each choice is right or wrong

A - Incorrect: A spiral can contain staged work, but sequential completion is not its defining driver.

B - Correct: Risk assessment and reduction guide each cycle’s work and further commitment.

C - Incorrect: Only delivering at the end is not the distinguishing spiral principle.

D - Incorrect: A risk-driven cycle can revise plans and requirements rather than rigidly freeze them.

Primary note proof

Study-note section SE06

> Spiral development is risk driven. Each cycle identifies objectives and alternatives, assesses and reduces important risks, develops/verifies an appropriate next result, and plans further commitment. A prototype may test a risky technology before a large investment. The amount and type of work depend on the risks.

Question sources

Software Engineering review questions · Question 22

Question 251 | NS0767

The V-Model derives its name from its shape. What does the BOTTOM of the 'V' represent?

A. Requirements Gathering

B. Deployment

C. Coding/Implementation

D. Maintenance

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Coding/implementation.

SE03 explicitly places coding at the bottom of the V while specification and design stages correspond to later testing levels. Requirements therefore begin the development side rather than occupy the bottom; deployment and maintenance are not that connecting implementation activity. This is a model-diagram convention, not a claim that all real work follows a perfect geometric sequence.

Why each choice is right or wrong

A - Incorrect: Requirements occur on the specification side, not at the coding bottom of the V.

B - Incorrect: Deployment is not the implementation activity joining the V-model's sides.

C - Correct: The note explicitly identifies coding as the bottom of the V.

D - Incorrect: Maintenance occurs in lifecycle evolution rather than at this implementation point.

Primary note proof

Study-note section SE03

> The V-model pairs specification/design stages with later test levels. Requirements inform acceptance tests; system design informs system tests; component/detail design informs integration/unit tests. Coding lies at the bottom. Test planning starts early. Stable requirements and strong assurance may favor this organization, but neither a V shape nor waterfall guarantees safety or correctness.

Question sources

Software Engineering review questions · Question 23

Question 252 | NS0768

Which organization standardizes and maintains UML as a modeling language?

A. IEEE

B. ISO

C. Object Management Group (OMG)

D. W3C

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Object Management Group (OMG)

The note identifies the Object Management Group as UML's standardizing and maintaining organization. UML is a modeling language rather than a software lifecycle process. This stewardship question does not claim that the organization originally invented every contributing notation. IEEE, ISO and W3C are not the organization to which the supplied note attributes this particular standardization and maintenance role.

Why each choice is right or wrong

A - Incorrect: IEEE is not the attributed UML steward.

B - Incorrect: ISO is not the organization attributed this role in the note.

C - Correct: OMG is the stated standardizing and maintaining organization.

D - Incorrect: W3C is not the attributed UML steward.

Primary note proof

Study-note section SE17

> A class box lists name, attributes and operations; + public, - private, # protected and ~ package indicate visibility. A class diagram shows static types and relationships, while an object diagram shows particular instances/links. UML is maintained and standardized by OMG; it is a modeling language, not a lifecycle process. Structural things describe conceptual or physical structural elements.

Question sources

Software Engineering review questions · Question 24

Question 253 | NS0769

In a UML Use Case Diagram, the rectangle enclosing all use cases is called:

A. System Context

B. System Boundary Box

C. Package Container

D. Module Frame

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: The system boundary box.

SE14 says actors are outside the system-boundary rectangle and use-case ovals are inside. The enclosing rectangle therefore marks the modeled system's boundary. A context may be communicated by the diagram, but it is not the specific box name requested; packages and modules serve different grouping roles rather than defining this external actor boundary.

Why each choice is right or wrong

A - Incorrect: System context is broader information, not the specific name of this enclosing boundary rectangle.

B - Correct: The rectangle separates the modeled system's use cases from external actor roles.

C - Incorrect: A package groups modeling elements but is not the stated system/actor boundary.

D - Incorrect: Module frame is not the use-case system-boundary term defined in SE14.

Primary note proof

Study-note section SE14

> Draw actors outside the system boundary rectangle and use-case ovals inside. Name a goal with a verb and noun. The diagram shows roles, goals and relationships, not the full detailed algorithm or UI layout. Being an internal component is different from being an external actor; the chosen boundary determines which role applies.

Question sources

Software Engineering review questions · Question 25

Question 254 | NS0770

When execution reaches an include relationship’s inclusion point in a base use case, what does include require?

A. Optional behavior that may be added to the base use case

B. The included behavior is required whenever execution reaches the inclusion point.

C. An inheritance relationship between two use cases

D. A timing constraint between two use cases

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: The included behavior is required whenever execution reaches the inclusion point.

Include factors required reused behavior into another use case. When the base reaches the inclusion point, the included behavior is incorporated and performed as specified. This does not imply that every possible conditional branch of the base reaches that point. Optional conditional additions correspond to extend, specialization to generalization, and timing constraints are separate from the inclusion relationship.

Why each choice is right or wrong

A - Incorrect: Conditional optional added behavior describes extend.

B - Correct: The included behavior is required when its inclusion point is reached.

C - Incorrect: Specialization is generalization rather than include.

D - Incorrect: Include is not a timing constraint.

Primary note proof

Study-note section SE16

> Generalization relates a specialized actor/use case to a more general one, with a hollow triangle toward the general element. An abstract use case specifies behavior for specialization and is not directly instantiated as a complete use case. It is not defined merely by reuse or lack of a direct actor line. Included behavior is incorporated where the inclusion is reached; do not infer every conditional branch in the base always executes.

Study-note section SE16

> Include factors required reused behavior: Submit Order includes Validate Order, with the dashed arrow from the including/base use case to the included one.

Question sources

Software Engineering review questions · Question 26

Question 255 | NS0771

In a UML class diagram, visibility marked with '#' indicates:

A. Public access

B. Private access

C. Protected access

D. Package-level access

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Protected visibility.

SE17 lists + public, - private, # protected and ~ package. The requested hash marker therefore means protected access. These symbols indicate visibility of a class member; they do not express object multiplicity, inheritance direction or class ownership, and the other choices correspond to different explicit markers.

Why each choice is right or wrong

A - Incorrect: Public visibility is marked with + rather than #.

B - Incorrect: Private visibility is marked with - rather than #.

C - Correct: The note explicitly assigns # to protected visibility.

D - Incorrect: Package visibility uses ~ rather than the protected # marker.

Primary note proof

Study-note section SE17

> A class box lists name, attributes and operations; + public, - private, # protected and ~ package indicate visibility. A class diagram shows static types and relationships, while an object diagram shows particular instances/links. UML is maintained and standardized by OMG; it is a modeling language, not a lifecycle process. Structural things describe conceptual or physical structural elements.

Question sources

Software Engineering review questions · Question 27

Question 256 | NS0772

In UML, which relationship is represented by a HOLLOW diamond shape pointing toward the 'whole' class?

A. Composition

B. Dependency

C. Aggregation

D. Generalization

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Shared aggregation.

SE18 defines shared aggregation with a hollow diamond at the whole end. Composition instead uses a filled diamond; dependency is a dashed arrow and generalization uses a hollow triangle. The question’s whole-end description is consistent with the notation, so the hollow shape uniquely identifies aggregation among these alternatives.

Why each choice is right or wrong

A - Incorrect: Composition uses a filled diamond, not the specified hollow diamond.

B - Incorrect: Dependency is represented by a dashed arrow rather than a whole-part diamond.

C - Correct: Shared aggregation uses a hollow diamond placed at the whole end.

D - Incorrect: Generalization uses a hollow triangle toward the general element rather than a diamond.

Primary note proof

Study-note section SE18

> Composition uses a filled diamond at the whole end and strong ownership: a part is in at most one composite at a time, and parts still owned are destroyed with that composite. Parts may be detached/transferred under a defined model. Thus “cannot ever exist independently” is an oversimplification. For Car composing Engine, the diamond belongs beside Car; saying it points to Engine is imprecise. The Car-Engine example describes composition.

Study-note section SE18

> Shared aggregation uses a hollow diamond at the whole end and weak whole-part meaning;

Question sources

Software Engineering review questions · Question 28

Question 257 | NS0773

Which statement accurately distinguishes UML composition from shared aggregation while allowing model-defined detachment or transfer?

A. Aggregation is for classes; Composition is for interfaces

B. Composition has exclusive composite ownership at a time and destroys still-owned parts with the whole; shared aggregation gives weaker whole-part meaning.

C. Aggregation implies inheritance; Composition does not

D. They are synonymous in modern UML

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: Composition has exclusive composite ownership at a time and destroys still-owned parts with the whole; shared aggregation gives weaker whole-part meaning.

Composition gives strong ownership: a part is in at most one composite at a time, and still-owned parts are destroyed with that composite. Shared aggregation indicates weaker whole-part meaning without that detailed lifecycle policy. A model can permit detachment or transfer, so permanent inability to exist independently is too strong. Neither relationship is inheritance, and composition is not restricted to interfaces.

Why each choice is right or wrong

A - Incorrect: Composition is not restricted to interfaces.

B - Correct: This states strong ownership and destruction of still-owned parts, unlike weak shared aggregation.

C - Incorrect: Aggregation is not inheritance; generalization is separate.

D - Incorrect: The ownership meanings differ, so they are not synonyms.

Primary note proof

Study-note section SE18

> Composition uses a filled diamond at the whole end and strong ownership: a part is in at most one composite at a time, and parts still owned are destroyed with that composite. Parts may be detached/transferred under a defined model. Thus "cannot ever exist independently" is an oversimplification. For Car composing Engine, the diamond belongs beside Car; saying it points to Engine is imprecise. The Car-Engine example describes composition.

Question sources

Software Engineering review questions · Question 29

Question 258 | NS0774

Under the course's sequence-diagram layout heuristic-rather than a mandatory UML column rule-which participant belongs in the second column?

A. The control object managing the use case

B. The actor who initiated the use case

C. A boundary object the actor used to start the use case

D. An entity object central to the use case

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: A boundary object the actor used to start the use case

The course's heuristic places the initiating actor first, the boundary object second, the control object third and entity objects afterward. The boundary used by the actor therefore occupies the second column under that convention. UML itself does not mandate this column order; lifeline semantics come from the modeled participants and messages rather than their horizontal position alone.

Why each choice is right or wrong

A - Incorrect: The heuristic places control third.

B - Incorrect: The initiating actor is first.

C - Correct: The actor-facing boundary is second under the specified heuristic.

D - Incorrect: Entities follow the boundary and control in this heuristic.

Primary note proof

Study-note section SE19

> One interaction-diagram layout heuristic puts the initiating actor first, boundary second, control third, then entities. This is not a required UML column order. In ReportEmergency, FieldOfficer activates ReportEmergencyButton, enters details through EmergencyReportForm, and later sees AcknowledgmentNotice; DispatcherStation belongs to the dispatcher side. Choose the button for initiating this particular scenario.

Question sources

Software Engineering review questions · Question 30

Question 259 | NS0775

In a UML Activity Diagram, a diamond shape is used to represent:

A. Initial State

B. Final State

C. Action Box

D. Decision Box

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer D: A decision node, also usable as a merge node in the appropriate flow.

SE20 names the guarded decision diamond and separately notes that the same shape can merge alternative flows. D is therefore a correct use of the shape, without claiming decision is its only role. Initial nodes are filled circles, actions are rounded, and final-state notation is not the diamond described.

Why each choice is right or wrong

A - Incorrect: The initial node is a filled circle rather than a diamond.

B - Incorrect: The diamond is not the standard final-state marker in the described notation.

C - Incorrect: Actions are rounded nodes rather than guarded-flow diamonds.

D - Correct: A diamond can represent a guarded decision; it can also merge alternative flows.

Primary note proof

Study-note section SE20

> An activity diagram models workflow: filled circle start, rounded actions, guarded decision diamond, thick fork/join bars and partitions/swimlanes for responsibility. A diamond can also merge alternative flows; a join synchronizes concurrent flows. A state machine models states and transitions triggered by events, often with guards/actions, for one object's lifecycle.

Question sources

Software Engineering review questions · Question 31

Question 260 | NS0776

In system design, 'coupling' between subsystems refers to:

A. How well objects within a subsystem work toward a common goal

B. The number of operations a subsystem exposes

C. The degree of dependency between two subsystems

D. The architectural pattern chosen for the system

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: The degree of dependency between subsystems.

SE22 explicitly distinguishes cohesion within a module from coupling between modules. A describes cohesion; C describes coupling. Interface operation count can affect design complexity but is not the definition of dependence, and an architectural pattern may influence coupling without being the relationship measure itself. The goal is usually low coupling with high cohesion.

Why each choice is right or wrong

A - Incorrect: Related purpose inside a subsystem defines cohesion rather than between-subsystem coupling.

B - Incorrect: Operation count alone does not measure the actual dependencies between subsystems.

C - Correct: Coupling is the degree of dependency between modules or subsystems.

D - Incorrect: An architectural style can affect coupling but is not its definition.

Primary note proof

Study-note section SE22

> Cohesion measures how well elements inside a module serve a related purpose; coupling measures dependency between modules. Seek high cohesion and low coupling. Shared mutable data and knowledge of another module’s internal representation increase coupling. Information hiding exposes stable operations while concealing changeable decisions.

Question sources

Software Engineering review questions · Question 32

Question 261 | NS0777

In architectural design, a 'closed architecture' in a layered system means:

A. No external access to any layer is allowed

B. A layer can access any layer at deeper levels freely

C. A layer can only call the layer immediately below it

D. All layers communicate through a central repository

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: Calls are restricted to the immediately lower layer.

SE23 defines a closed layered architecture as allowing calls only to the immediately lower layer. Open layering may permit approved skipping, so B describes a different policy. Closed does not mean no external access or a shared repository; it concerns permitted dependencies between layers rather than the system's exposure or storage organization.

Why each choice is right or wrong

A - Incorrect: Closed layering restricts interlayer calls, not every form of external access.

B - Incorrect: Skipping freely to deeper layers contradicts the immediate-lower-layer restriction.

C - Correct: A layer calls only the immediately lower layer in the defined closed architecture.

D - Incorrect: A central repository is another architectural arrangement and does not define closed layering.

Primary note proof

Study-note section SE23

> A layer is a logical abstraction level; a tier is a deployment separation. A closed layered architecture permits calls only to the immediately lower layer; an open one permits approved skipping. For top-to-bottom A,B,C,D, A→B obeys closed layering, A→C skips, and lower-to-upper calls violate the simple dependency rule unless explicitly mediated by a defined mechanism.

Question sources

Software Engineering review questions · Question 33

Question 262 | NS0778

Which architectural style divides subsystems into Model, View, and Controller components?

A. Repository Style

B. Client/Server Style

C. Peer-to-Peer Style

D. Model-View-Controller Style

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer D: Model-View-Controller.

SE24 explicitly defines MVC as separating a domain model, display view and interaction controller. D names that exact style. Repository, client-server and peer-to-peer organize central data or interacting roles differently; any may be combined with MVC, but none is the name of this particular three-responsibility separation.

Why each choice is right or wrong

A - Incorrect: Repository style centers shared data rather than the specific model/view/controller separation.

B - Incorrect: Client-server separates requesters and service providers, not these three presentation-interaction responsibilities.

C - Incorrect: Peer-to-peer allows participants to play requester and provider roles rather than defining MVC components.

D - Correct: Model-View-Controller explicitly names the three responsibilities in the question.

Primary note proof

Study-note section SE24

> Repository style shares central data, simplifying common access but creating bottleneck/schema dependencies. MVC separates domain model, display view and interaction controller. Client-server separates requesters from service providers; peer-to-peer permits both roles. Pipe-and-filter applies successive transformations to data streams. Three-tier separates presentation, application logic and storage; some four-tier variants split presentation client/server.

Question sources

Software Engineering review questions · Question 34

Question 263 | NS0779

In the Three-Tier architectural style, which layer handles 'storing, retrieving, and querying persistent data'?

A. Interface Layer

B. Application Logic Layer

C. Storage Layer

D. Presentation Layer

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer C: The storage layer/tier.

SE24 separates three-tier presentation, application logic and storage. Persistent data storage and retrieval belong to the storage responsibility, whereas presentation handles interaction and application logic coordinates domain behavior. The question uses layer for a tier-style role; that terminology does not change which responsibility owns persistence.

Why each choice is right or wrong

A - Incorrect: The interface responsibility handles interaction rather than persistent storage and querying.

B - Incorrect: Application logic coordinates domain operations but delegates persistence to the storage responsibility.

C - Correct: The storage role handles persistent data, including retrieval and queries.

D - Incorrect: Presentation formats user interaction and output rather than owning persistent data storage.

Primary note proof

Study-note section SE24

> Repository style shares central data, simplifying common access but creating bottleneck/schema dependencies. MVC separates domain model, display view and interaction controller. Client-server separates requesters from service providers; peer-to-peer permits both roles. Pipe-and-filter applies successive transformations to data streams. Three-tier separates presentation, application logic and storage; some four-tier variants split presentation client/server.

Study-note section SE23

> A layer is a logical abstraction level; a tier is a deployment separation. A closed layered architecture permits calls only to the immediately lower layer; an open one permits approved skipping. For top-to-bottom A,B,C,D, A→B obeys closed layering, A→C skips, and lower-to-upper calls violate the simple dependency rule unless explicitly mediated by a defined mechanism.

Question sources

Software Engineering review questions · Question 35

Question 264 | NS0780

In the analysis model, which type of object represents the bridges between the system and the actors?

A. Entity Objects

B. Control Objects

C. Boundary Objects

D. Interface Objects

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Boundary objects.

SE13 defines boundary objects as mediating interaction, with ReportForm as an example. That is the bridge between external actors and the system. Entity objects hold domain information, while controls coordinate use cases. Interface objects is a generic descriptive phrase, but boundary is the explicitly defined analysis-category name requested among these choices.

Why each choice is right or wrong

A - Incorrect: Entity objects represent domain information rather than the actor-facing interaction bridge.

B - Incorrect: Control objects coordinate use-case behavior rather than primarily mediate actor interaction.

C - Correct: Boundary objects are explicitly defined as mediating system-actor interaction.

D - Incorrect: Interface objects is not the named entity/boundary/control category specified here's analysis model.

Primary note proof

Study-note section SE13

> A scenario is one concrete execution; a use case abstracts the related successful and alternative scenarios. Analysis integrates functional (use cases/scenarios), structural/object (classes) and dynamic (states/interactions) models. Entity objects represent domain information such as Incident; boundary objects mediate interaction, such as ReportForm; control objects coordinate a use case. A UI form is normally boundary, not entity.

Question sources

Software Engineering review questions · Question 36

Question 265 | NS0781

Which heuristic rule states that there should be 'one control object per actor' in a use case?

A. CRC Card Heuristic

B. Control Object Identification Heuristic

C. Boundary Object Heuristic

D. Sequence Diagram Column Rule

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: The control-object identification heuristic.

SE13 permits one control per use case or participating actor as a useful starting point, while denying a mandatory UML rule. The stem already asks for a heuristic, so B correctly identifies the control-object selection guideline. CRC, boundary identification and column layout concern different tasks; none makes the count a required lifetime or architecture constraint.

Why each choice is right or wrong

A - Incorrect: CRC records class responsibilities and collaborators rather than prescribing this candidate-control count.

B - Correct: This is the heuristic for identifying candidate use-case control objects, not a universal UML constraint.

C - Incorrect: Boundary heuristics identify interaction-facing objects rather than coordinating controls.

D - Incorrect: Sequence-column layout orders participants visually and does not define candidate control-object counts.

Primary note proof

Study-note section SE13

> Use nouns as candidate objects, possessive descriptions as candidate attributes, and verbs such as manages/reports-to as candidate associations. These are heuristics, not automatic parsing rules. CRC means Class, Responsibilities, Collaborators. A control per use case or participating actor can be a useful starting point, not a mandatory UML rule or fixed object lifetime.

Question sources

Software Engineering review questions · Question 37

Question 266 | NS0782

CRC cards stand for:

A. Class, Requirements, Conditions

B. Component, Relationship, Constraints

C. Class, Responsibilities, Collaborators

D. Control, Reuse, Composition

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: Class, Responsibilities, Collaborators.

SE13 expands CRC directly as Class, Responsibilities, Collaborators. The card identifies the class, what it must do or know, and which other objects help fulfill those responsibilities. The distractors replace these terms with requirements, components or unrelated design concepts and therefore do not preserve the stated expansion.

Why each choice is right or wrong

A - Incorrect: Requirements and Conditions are not the R and final C in the stated expansion.

B - Incorrect: Component, Relationship and Constraints do not match the class-card terminology.

C - Correct: This is the exact CRC expansion supplied in SE13.

D - Incorrect: Control, Reuse and Composition are different concepts rather than the CRC card fields.

Primary note proof

Study-note section SE13

> Use nouns as candidate objects, possessive descriptions as candidate attributes, and verbs such as manages/reports-to as candidate associations. These are heuristics, not automatic parsing rules. CRC means Class, Responsibilities, Collaborators. A control per use case or participating actor can be a useful starting point, not a mandatory UML rule or fixed object lifetime.

Question sources

Software Engineering review questions · Question 38

Question 267 | NS0783

In UML association multiplicity, what does '0..*' mean?

A. Exactly one instance

B. Zero or exactly one instance

C. Zero or more instances

D. One or more instances

Hint

Match the information to be represented with each model’s purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Zero or more instances.

SE17 defines 0..* as any nonnegative number. Zero is allowed and the star places no finite upper bound, so C fits. The alternatives correspond to 1, 0..1 and 1..* respectively. Multiplicity at one end is interpreted relative to one object at the opposite end, not as a whole-system object count.

Why each choice is right or wrong

A - Incorrect: Exactly one corresponds to multiplicity 1 rather than 0..*.

B - Incorrect: Optional single-valued multiplicity is 0..1, not an unbounded star.

C - Correct: Zero is permitted and the upper star allows any number of related instances.

D - Incorrect: One or more requires lower bound 1, written 1..*.

Primary note proof

Study-note section SE17

> Multiplicity at one end counts how many objects there may correspond to one object at the opposite end: 1 exactly one, 0..1 optional, 0..* any nonnegative number, 1..* at least one. A one-to-many association does not automatically mean composition. Read each end separately before choosing cardinality.

Question sources

Software Engineering review questions · Question 39

Question 268 | NS0784

In Abbott's heuristics for identifying associations, which phrase pattern most directly suggests a relationship between classes?

A. Look for adjective phrases describing class properties

B. Examine verb phrases like 'has', 'is part of', 'manages', 'reports to'

C. Find all nouns that appear more than twice

D. Identify all user interface terms

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer B: Verb phrases expressing relationships, such as manages or reports to.

SE13 recommends nouns as candidate objects and verbs such as manages/reports-to as candidate associations. B is the relationship-phrase option. Adjectives generally describe properties, repeated nouns do not automatically imply an association, and UI terms suggest boundary concerns. These are candidate-identification heuristics rather than a proof that every verb must become a modeled link.

Why each choice is right or wrong

A - Incorrect: Property descriptions suggest attributes rather than directly identifying links between classes.

B - Correct: Relationship verbs are the note’s direct heuristic for candidate associations.

C - Incorrect: Noun frequency is not the stated association-identification rule; nouns instead suggest candidate objects.

D - Incorrect: UI terminology may suggest boundaries but does not directly establish an association between domain classes.

Primary note proof

Study-note section SE13

> Use nouns as candidate objects, possessive descriptions as candidate attributes, and verbs such as manages/reports-to as candidate associations. These are heuristics, not automatic parsing rules. CRC means Class, Responsibilities, Collaborators. A control per use case or participating actor can be a useful starting point, not a mandatory UML rule or fixed object lifetime.

Question sources

Software Engineering review questions · Question 40

Question 269 | NS0785

In software testing, which type of testing evaluates the software's behavior WITHOUT knowledge of its internal code structure?

A. White Box Testing

B. Grey Box Testing

C. Black Box Testing

D. Structural Testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer C: Black-box testing.

SE36 contrasts black-box cases derived from required behavior with white-box cases derived from implementation structure. Grey-box uses partial internal knowledge. The question excludes internal-code knowledge, so black-box is the appropriate technique. Structural testing instead refers to the very internal paths or statements that this scenario does not inspect.

Why each choice is right or wrong

A - Incorrect: White-box testing explicitly uses internal implementation structure to derive cases.

B - Incorrect: Grey-box testing uses partial internal knowledge rather than excluding it.

C - Correct: Black-box testing derives cases from externally specified behavior without needing code structure.

D - Incorrect: Structural testing is based on internal control or code structure, contrary to the stem.

Primary note proof

Study-note section SE36

> Black-box techniques derive cases from specified behavior; white-box techniques derive them from internal structure; grey-box uses partial internal knowledge. Regression testing checks for unintended effects after change; confirmation testing checks the particular defect repair. Exploratory testing combines learning, design and execution and is experience based. Manual/automated describe execution support, not a compulsory black-/white-box classification.

Question sources

Software Engineering review questions · Question 41

Question 270 | NS0786

Which type of functional testing verifies that individual components work correctly in ISOLATION?

A. System Testing

B. Integration Testing

C. Regression Testing

D. Unit Testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer D: Unit testing.

SE35 defines unit testing as isolating a function, class or small component, often replacing dependencies with stubs or mocks. Integration checks combined interactions and system testing evaluates the whole system. Regression describes testing after changes and can occur at different levels, so it is not the isolation level named by the question.

Why each choice is right or wrong

A - Incorrect: System testing evaluates the complete integrated system rather than an isolated component.

B - Incorrect: Integration testing focuses on interactions among combined parts, not isolation.

C - Incorrect: Regression checks unintended effects after change and is not inherently a component-isolation level.

D - Correct: Unit testing evaluates a small component in isolation, with dependencies replaceable by test doubles.

Primary note proof

Study-note section SE35

> Unit testing isolates a function, class or small component, often replacing dependencies with stubs/mocks; a driver invokes the unit and a stub substitutes a called dependency. Integration testing exercises interfaces and interactions among combined parts. System testing checks the complete integrated system against functional and nonfunctional requirements in a representative environment.

Question sources

Software Engineering review questions · Question 42

Question 271 | NS0787

Non-functional testing that evaluates how the system behaves under expected workloads and measures response speed is called:

A. Usability Testing

B. Reliability Testing

C. Performance Testing

D. Compatibility Testing

Hint

Separate testing scope, technique and purpose. Use the stated workload, observable result or coverage target rather than a universal ranking.

Answer C: Performance testing, specifically load-oriented testing under expected demand.

SE36 defines performance tests as assessing speed, throughput and resource use under a specified workload and distinguishes expected load from stress beyond normal limits. The stem explicitly adds workload and response speed, so it supports performance testing rather than merely a vague normal-condition test. Usability, reliability and compatibility target different quality questions.

Why each choice is right or wrong

A - Incorrect: Usability focuses on task success and ease of use rather than response-speed measurements under load.

B - Incorrect: Reliability concerns failure-free service, not primarily the stated response-time criterion.

C - Correct: Speed under an expected workload is performance/load testing as defined in SE36.

D - Incorrect: Compatibility examines cooperating environments rather than the specified speed-under-load behavior.

Primary note proof

Study-note section SE36

> Functional tests check required behavior against the SRS and acceptance criteria. Performance tests assess speed, throughput and resources under a specified workload; load uses expected demands and stress exceeds normal limits. Usability, security, compatibility and reliability tests target their respective qualities. “Normal operating conditions” alone does not distinguish functional from performance testing.

Question sources

Software Engineering review questions · Question 43

Question 272 | NS0788

In the analysis phase, which model type uses use cases and scenarios to describe what the software should do?

A. Dynamic Model

B. Analysis Object Model

C. Functional Model

D. Structural Model

Hint

Distinguish observable services, measurable qualities and implementation constraints. Check scope, stakeholder meaning and the requested level of specificity.

Answer C: The functional model.

SE13 explicitly associates use cases and scenarios with functional modeling, classes with structural/object modeling, and states/interactions with dynamic modeling. Thus C names the requested description of what the system should do. The alternatives describe state-dependent behavior or static object structure rather than this user-goal and scenario view.

Why each choice is right or wrong

A - Incorrect: Dynamic models emphasize states and interactions rather than the named use-case/scenario category.

B - Incorrect: The analysis object model captures classes and relationships rather than the functional use-case view.

C - Correct: Use cases and scenarios are explicitly the functional analysis model.

D - Incorrect: Structural modeling describes static classes and links rather than required user-goal flows.

Primary note proof

Study-note section SE13

> A scenario is one concrete execution; a use case abstracts the related successful and alternative scenarios. Analysis integrates functional (use cases/scenarios), structural/object (classes) and dynamic (states/interactions) models. Entity objects represent domain information such as Incident; boundary objects mediate interaction, such as ReportForm; control objects coordinate a use case. A UI form is normally boundary, not entity.

Question sources

Software Engineering review questions · Question 44

Question 273 | NS0789

In an analysis model, an object coordinates a use case and moves among Idle, WaitingForReply and Completing states as events arrive. Domain entities are passive records and the form has no persistent modes in this scenario. Which object category is the direct state-machine target described?

A. Entity Objects

B. Boundary Objects

C. Control Objects

D. All objects equally

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Control Objects

A control object coordinates a use case, and the described coordinator has significant event-dependent states. A state machine is therefore directly useful for that control object in this scenario. Entities and boundaries can also have meaningful lifecycles in other systems, but here their stated behavior does not supply the modeled states. Suitability follows from responsibilities and lifecycle behavior, not equal need across all objects.

Why each choice is right or wrong

A - Incorrect: The scenario's entities are passive records rather than the stated coordinator.

B - Incorrect: The form has no persistent modes and is not the use-case coordinator.

C - Correct: The stateful coordinator is a control object.

D - Incorrect: The need depends on significant state behavior, not equality across every object.

Primary note proof

Study-note section SE20

> Choose a sequence diagram for message order between participants, a state machine for event-dependent state changes, and activity for workflow. Both sequence and state diagrams describe behavior over time. State machines are useful for objects with significant states, whether entity or control. A DFD shows data stores/flows and transformations from inputs to outputs; it is not a UML diagram.

Study-note section SE13

> A scenario is one concrete execution; a use case abstracts the related successful and alternative scenarios. Analysis integrates functional (use cases/scenarios), structural/object (classes) and dynamic (states/interactions) models. Entity objects represent domain information such as Incident; boundary objects mediate interaction, such as ReportForm; control objects coordinate a use case. A UI form is normally boundary, not entity.

Question sources

Software Engineering review questions · Question 45

Question 274 | NS0790

In the ReportEmergency use case example, which boundary object allows the FieldOfficer to initiate the reporting process?

A. EmergencyReportForm

B. AcknowledgmentNotice

C. ReportEmergencyButton

D. DispatcherStation

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: ReportEmergencyButton.

SE19 explicitly says FieldOfficer activates ReportEmergencyButton, supplies details through EmergencyReportForm and later sees AcknowledgmentNotice. DispatcherStation belongs to the dispatcher side. Therefore the button initiates this particular scenario; the form handles later input and the notice reports completion. This is grounded in the named example rather than a generic rule that every use case starts with a button.

Why each choice is right or wrong

A - Incorrect: The form receives incident details after initiation rather than starting the documented scenario.

B - Incorrect: The acknowledgment is seen later as feedback, not used to initiate reporting.

C - Correct: The note explicitly identifies this button as the FieldOfficer's initiating boundary object.

D - Incorrect: DispatcherStation is on the dispatcher side rather than the FieldOfficer's initiation interface.

Primary note proof

Study-note section SE19

> One interaction-diagram layout heuristic puts the initiating actor first, boundary second, control third, then entities. This is not a required UML column order. In ReportEmergency, FieldOfficer activates ReportEmergencyButton, enters details through EmergencyReportForm, and later sees AcknowledgmentNotice; DispatcherStation belongs to the dispatcher side. Choose the button for initiating this particular scenario.

Question sources

Software Engineering review questions · Question 46

Question 275 | NS0791

Generalization in UML modeling is best described as:

A. Dividing a class into multiple subclasses with no shared properties

B. Identifying a general concept where other concepts are more specific versions of it

C. Creating interfaces for external system communication

D. Grouping unrelated classes into a package for organization

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer B: A general concept with more-specific specializations.

SE16 relates a specialized actor or use case to a more general one; SE18 gives the corresponding class generalization toward a superclass. B expresses that abstraction/specialization relationship. It does not merely package unrelated classes, create an external interface or split classes while denying shared general properties.

Why each choice is right or wrong

A - Incorrect: Specialization is related through a general concept rather than requiring no shared properties.

B - Correct: Generalization identifies the common general element of which other concepts are more specific versions.

C - Incorrect: External communication interfaces do not by themselves define a generalization relationship.

D - Incorrect: Packaging unrelated classes is organizational grouping rather than specialization of a shared general concept.

Primary note proof

Study-note section SE18

> Association connects related instances. A dashed dependency arrow means a client uses a supplier. Generalization uses a solid line and hollow triangle pointing to the superclass/general element. Shared aggregation uses a hollow diamond at the whole end and weak whole-part meaning; it does not by itself define a detailed lifecycle policy.

Study-note section SE16

> Generalization relates a specialized actor/use case to a more general one, with a hollow triangle toward the general element. An abstract use case specifies behavior for specialization and is not directly instantiated as a complete use case. It is not defined merely by reuse or lack of a direct actor line. Included behavior is incorporated where the inclusion is reached; do not infer every conditional branch in the base always executes.

Question sources

Software Engineering review questions · Question 47

Question 276 | NS0792

The 'high cohesion, low coupling' principle in subsystem design means:

A. Subsystems should be tightly interconnected for maximum communication efficiency

B. Objects within a subsystem should be closely related, while subsystems themselves should be mostly independent

C. Each subsystem should have minimal internal functionality to reduce complexity

D. All subsystems should share a single central data repository

Hint

Separate responsibilities inside a component from dependencies across boundaries. Evaluate the stated structure, change and reuse goals without assuming guaranteed benefits.

Answer B: Related responsibilities within a subsystem, limited dependencies between subsystems.

SE22 defines cohesion as related purpose within a module and coupling as dependency across modules, then recommends high cohesion and low coupling. B states both dimensions. It does not require minimal functionality, tight cross-connections or one shared repository; those choices can undermine coherent responsibilities or increase dependency rather than establish the principle.

Why each choice is right or wrong

A - Incorrect: Tight interconnection increases coupling rather than meeting the low-coupling goal.

B - Correct: Closely related internal responsibilities and limited external dependencies express both parts of the principle.

C - Incorrect: Cohesion concerns related purpose, not simply having as little functionality as possible.

D - Incorrect: A shared repository is a style choice and can create dependencies; it does not define cohesion or coupling.

Primary note proof

Study-note section SE22

> Cohesion measures how well elements inside a module serve a related purpose; coupling measures dependency between modules. Seek high cohesion and low coupling. Shared mutable data and knowledge of another module’s internal representation increase coupling. Information hiding exposes stable operations while concealing changeable decisions.

Question sources

Software Engineering review questions · Question 48

Question 277 | NS0793

A safety-critical contract requires traceable assurance records and reviewed verification before release. Waterfall, V-model and spiral proposals include those obligations; the proposed lightweight Agile variant supplies working code but omits the required assurance records. Which proposal is least appropriate as specified?

A. Waterfall Model

B. V-Model

C. Agile Model

D. Spiral Model

Hint

Identify whether the scenario prioritizes delivery, learning, adaptation or risk reduction. Several processes can share individual practices, so compare their defining emphasis.

Answer C: Agile Model

The specified Agile proposal omits mandatory assurance records, so it cannot satisfy the stated contract without modification. The other proposals include those obligations and are not excluded on that ground. This is a judgment about the supplied lightweight variant, not Agile in general: the note explicitly allows Agile to incorporate required documentation and assurance. Stable requirements alone would not establish a universal ranking.

Why each choice is right or wrong

A - Incorrect: This proposal includes the required reviewed deliverables.

B - Incorrect: This proposal includes the required assurance obligations.

C - Correct: The specified lightweight variant omits mandatory records and is therefore least suitable as proposed.

D - Incorrect: This proposal includes the stated assurance obligations.

Primary note proof

Study-note section SE05

> Scrum uses timeboxed Sprints of at most one month, with a Sprint Goal, selected work and a usable Increment meeting a Definition of Done. Scope is discussed before and during planning; it is not forbidden to agree scope. Timeboxes tend to stay fixed while content is adapted without endangering the goal. Agile can be combined with required assurance/documentation; safety-critical context alone cannot prove it is unsuitable.

Study-note section SE03

> The V-model pairs specification/design stages with later test levels. Requirements inform acceptance tests; system design informs system tests; component/detail design informs integration/unit tests. Coding lies at the bottom. Test planning starts early. Stable requirements

and strong assurance may favor this organization, but neither a V shape nor waterfall guarantees safety or correctness.

Study-note section SE03

Classical waterfall separates planned phases with reviewed deliverables. It fits well-understood, stable requirements and familiar technology; late feedback makes changes costly.

Question sources

Software Engineering review questions · Question 49

Question 278 | NS0794

In a sequence diagram, what does an 'X' (cross) at the end of a lifeline dashed line indicate?

A. The object is temporarily inactive

B. The object is waiting for a message

C. The object ceases to exist at that point

D. The object is calling itself recursively

Hint

Match the information to be represented with each model's purpose. Read relationship direction, boundary and ownership conditions before classifying it.

Answer C: Modeled destruction of the object at that point.

SE19 explicitly identifies X as modeled destruction, so C is correct in UML diagram semantics. Temporary inactivity and waiting do not end an object's modeled existence, and a recursive message is represented by an interaction rather than destruction. The marker does not guarantee the runtime moment at which a garbage collector reclaims memory.

Why each choice is right or wrong

A - Incorrect: Temporary inactivity does not terminate the object's lifeline.

B - Incorrect: Waiting for a message does not mean the modeled object has been destroyed.

C - Correct: X marks the object's modeled destruction at that point, not necessarily immediate memory reclamation.

D - Incorrect: A recursive call is an interaction message and does not use the destruction marker.

Primary note proof

Study-note section SE19

> Sequence diagrams show interactions across lifelines, with order/time generally downward. Messages cross between participants; activation rectangles show execution. A creation message starts a lifeline; an X marks modeled destruction, not a guarantee about when a garbage collector runs. Synchronous calls wait for completion; asynchronous messages permit the sender to proceed.

Question sources

Software Engineering review questions · Question 50

Design and Analysis of Algorithms

Questions & Worked Answers

141 questions + 4 source variants

Question 1 | Q00065

Which one of the following is not an example of divide and conquer algorithm?

A. Quick sort

B. Binary searching

C. Merge sort

D. Heap sort

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: Heap sort

DAA16 explicitly classifies heap sort as outside standard divide and conquer: it builds a heap and repeatedly extracts its maximum. Quicksort partitions and recursively sorts ranges; merge sort recursively sorts halves; binary search recursively or iteratively reduces to a smaller half. Thus heap sort is the single exclusion among the listed algorithms.

Why each choice is right or wrong

A - Incorrect: Quicksort partitions its input and recursively sorts the resulting smaller ranges, a standard divide-and-conquer pattern.

B - Incorrect: Binary search repeatedly reduces a sorted search interval to one half; the note treats this as divide-and-conquer.

C - Incorrect: Merge sort divides into halves, sorts each recursively, and combines them by merging.

D - Correct: Heap sort builds a heap and repeatedly restores it after extraction, rather than independently sorting and combining subinstances.

Primary note proof

Study-note section DAA12

> Divide and conquer solves smaller largely independent instances and combines results. DP reuses a defined set of subproblem states. Greedy commits to a feasible local choice only when a safety argument supports it. Backtracking grows/retracts candidates when constraints fail; branch and bound also rejects candidates whose best possible objective cannot beat an incumbent.

Study-note section DAA16

Heap sort builds a max-heap in Θ(n), then extracts/restores in O(log n) per item: O(n log n) total, Θ(1) auxiliary space, ordinarily unstable. It is not standard D&C. Merge sort offers stability/worst-case guarantees; quicksort's practical advantage depends on implementation/workload.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 17 | lines 23-29

• Breaking large problems into smaller subproblems • Example of algorithms uses divide-and-conquer strategy : • Binary search, • Finding maximum and minimum • Merge sort • Quick sort • selection sort

Design and Analysis of Algorithms module | PDF page 23 | lines 19-29

QUICK SORT • The divide-and-conquer approach can be used to arrive at an efficient sorting method different from merge sort. • In merge sort, the file a[1:n] was divided at its midpoint into sub arrays which were independently sorted & later merged. • In Quick sort, the division into 2 sub arrays is made so that the sorted sub arrays do not need to be merged later. • This is accomplished by rearranging the elements in a[1:n] such that a[I]<=a[j] for all I between 1 & n and all j between (m+1) & n for some m, 1<=m<=n. • Thus the elements in a[1:m] & a[m+1:n] can be independently sorted. • No merge is needed. This rearranging is referred to as partitioning.

Design and Analysis of Algorithms module | PDF page 21 | lines 18-29

MERGE SORT • As another example divide-and-conquer, we investigate a sorting algorithm that has the nice property that is the worst case its complexity is O(n log n) • This algorithm is called merge sort • We assume throughout that the elements are to be sorted in non-decreasing order. • Given a sequence of ‗n' elements a[1],…,a[n] the general idea is to imagine then split into 2 sets a[1],…..,a[n/2] and a[[n/2]+1],….a[n]. • Each set is individually sorted, and the resulting sorted sequences are merged to produce a single sorted sequence of ‗n' elements. • Thus, we have another ideal example of the divide-and-conquer strategy in which the splitting is into 2 equal-sized sets & the combining operation is the merging of 2 sorted sets into one.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 65

Question 2 | Q00149

For n≥2, determine the tight asymptotic running time of this syntax-repaired C++-style pseudocode under mathematical-integer, unit-cost arithmetic and output:
function(n) { a=0; print("Enter a positive integer"); for(i=1; i<n; i*=2) a=i; return 0; }

A. Θ(n log n)

B. $\Theta(n^2)$

C. Θ(log n)

D. Θ(n)

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C: Θ(log n)

The loop visits i=1,2,4,... while i<n, so it executes ceil($\log_2 n$) iterations. Each iteration has constant work under the stated arithmetic and output model, and initialization, the fixed prompt and return add constant overhead. The total is therefore Θ(log n).

Why each choice is right or wrong

A - Incorrect: n log n grows faster and is not the tight order.

B - Incorrect: Quadratic grows faster and is not the tight order.

C - Correct: The doubling loop takes logarithmically many constant-cost iterations.

D - Incorrect: Linear is a looser upper bound rather than the tight order.

Primary note proof

Study-note section DAA06

> Sequential costs add; independent nested loop counts multiply. A dependent loop requires summation. n iterations of a constant body give Θ(n); ∑i=1.. n i=n(n+1)/2 gives Θ(n^2). Doubling i=1,2,4,... until i≥n takes ceil(log 2 n) iterations for n>1. A 10n loop is linear under constant-cost output.

Study-note section DAA02

> Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 10 | lines 6-20

> Analysis Rules: 1. We assume an arbitrary time unit. 2. Execution of one of the following operations takes time 1: • Assignment Operation • Single Input/output Operation • Single Boolean Operations • Single Arithmetic Operations • Function Return 3. Running time of a selection statement (if, switch) is the time for the condition evaluation plus the maximum of the running times for the individual clauses in the selection. 4. Loops: Running time for a loop is equal to the running time for the statements inside the loop multiplied by the number of iterations. The total running time of a statement inside a group of nested loops is the running time of the statements multiplied by the product of the sizes of all the loops. For nested loops, analyze inside out.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching • Works on ordered/sorted list • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list • The computational time for this algorithm is proportional to log2 n.

Question sources

Addis Ababa University mock exam · Question 51

Question 3 | Q00173

Recursion is a method in which the solution of a problem depends on:

A. Larger instances of the same problem

B. Smaller instances of the same problem

C. Larger instances of different problems

D. Smaller instances of different problems

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: Smaller instances of the same problem

Recursion reduces a problem to simpler instances of that same problem (PDF p.14). Thus smaller same-problem instances, with a terminating base case, match the definition.

Why each choice is right or wrong

A - Incorrect: Larger instances do not implement the reducing step on p.14.

B - Correct: Smaller instances preserve the problem while reducing its size (p.14).

C - Incorrect: Changing both size upward and problem identity fails the recursive reduction (p.14).

D - Incorrect: Smaller instances of a different problem are not the same-problem recursion described on p.14.

Primary note proof

Study-note section DAA09

Include every recursive call, nonrecursive work and base case. Expansion gives $T(n)=T(n-1)+\Theta(1)\rightarrow\Theta(n)$, and $T(n)=T(n/2)+\Theta(1)\rightarrow\Theta(\log n)$. For $2T(n/2)+\Theta(n)$, logarithmically many levels each cost $\Theta(n)$, giving $\Theta(n \log n)$.

Study-note section DAA01

Prove an invariant initially, after each step and at termination; for recursion prove base cases and the inductive reduction. Testing finds failures but finite tests do not prove correctness for all inputs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 30-33

> Recursion is a particularly powerful kind of reduction, which can be described loosely as follows: • If the given instance of the problem is small or simple enough, just solve it. • Otherwise, reduce the problem to one or more simpler instances of the same problem.

Question sources

Addis Ababa University mock exam · Question 75

Computer Science Exit Exam · Sene 2016 · Question 16-3

Computer Science Exit Exam practice collection · Question 96

Question 4 | Q00175

If an optimal solution can be created for a problem by constructing optimal solutions for its subproblems, the problem possesses property.

A. Overlapping subproblems

B. Optimal substructure

C. Memoization

D. Greedy

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: Optimal substructure

Optimal substructure means an optimal solution contains optimal solutions to subproblems (PDF p.28). The principle of optimality on p.49 expresses exactly the construction in the stem.

Why each choice is right or wrong

A - Incorrect: Overlap concerns repeated subproblems, not optimality of their solutions (pp.28,49).

B - Correct: This is the named property defined on p.28.

C - Incorrect: Memoization stores computed answers; it is not the optimality property (p.49).

D - Incorrect: Greedy is a design strategy requiring additional choice conditions (p.28).

Primary note proof

Study-note section DAA25

> Define state meaning, choices, recurrence, bases and dependency order. Prove replacing a subsolution by a better one improves the full solution; otherwise state information may be

missing. Show the recurrence covers all legal final choices and generates feasible candidates, then induct in dependency order. Store choices for reconstruction.

Study-note section DAA19

A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it. Optimal substructure alone does not prove this safe. An exchange proof inserts the greedy choice into an optimum without worsening cost; recurse. MST cut proofs use this exchange.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 28 | lines 3-4

• Optimal substructure: An optimal solution to the problem contains an optimal solution to sub problems.

Design and Analysis of Algorithms module | PDF page 49 | lines 2-12

The Principle of Optimality The dynamic programming relies on a principle of optimality. This principle states that in an optimal sequence of decisions or choices, each subsequence must also be optimal. For example, in matrix chain multiplication problem, not only the value we are interested in is optimal but all the other entries in the table are also represent optimal. The principle can be related as follows: the optimal solution to a problem is a combination of optimal solutions to some of its sub problems. The difficulty in turning the principle of optimally into an algorithm is that it is not usually obvious which sub problems are relevant to the problem under consideration. General method-multistage graphs-all pair shortest path algorithm-Optimal Binary Search trees- 0/1 knapsack Problem-Reliability design-traveling salesman problem

Question sources

Addis Ababa University mock exam · Question 77

Question 5 | Q00176

If a problem can be broken into subproblems that are reused several times, the problem possesses property.

A. Overlapping subproblems

B. Optimal substructure

C. Memoization

D. Greedy

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: Overlapping subproblems

Reusing a subproblem several times means subproblems share sub-subproblems, the contrast made on PDF p.48. This is overlapping subproblems; storing an answer is a response to that property, not the property itself.

Why each choice is right or wrong

A - Correct: Shared sub-subproblems and repeated work are described on p.48.

B - Incorrect: Optimal substructure concerns optimal solutions rather than repetition (p.28).

C - Incorrect: Saving results in a table addresses repeated work; it is not the reuse property (p.48).

D - Incorrect: A greedy choice is unrelated to whether subproblems recur (pp.28,48).

Primary note proof

Study-note section DAA25

> Overlapping states motivate reuse; optimal substructure supports optimization. These are textbook signatures, not requirements that every DP problem optimize or every state have multiple users. DP also counts/decides. Cost=states×work per state and can remain exponential.

Study-note section DAA26

> Memoization recursively checks a cache before solving a state and stores its result. Tabulation orders dependencies explicitly; chosen domains may exclude irrelevant states. Cache keys include all relevant parameters; recursion/locality/sparsity affect implementation cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

> Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Design and Analysis of Algorithms module | PDF page 28 | lines 3-4

> • Optimal substructure: An optimal solution to the problem contains an optimal solution to sub problems.

Question sources

Addis Ababa University mock exam · Question 78

Question 6 | Q00177

If a problem can be solved by combining optimal solutions to non-overlapping sub-problems, the strategy is called _____.

A. Dynamic programming

B. Divide and Conquer

C. Recursion

D. Greedy

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: Divide and Conquer

PDF p.48 explicitly contrasts independent subproblems in divide and conquer with shared sub-subproblems in dynamic programming. Combining independent solutions therefore identifies divide and conquer; optimality alone does not identify greedy.

Why each choice is right or wrong

A - Incorrect: Dynamic programming is motivated by non-independent shared subproblems (p.48).

B - Correct: Independent subproblems are recursively solved and combined (pp.17,48).

C - Incorrect: Recursion is a mechanism; the more specific independent-split-and-combine strategy is given on p.48.

D - Incorrect: Greedy needs a locally justified choice, which this stem does not specify (p.48 contrast).

Primary note proof

Study-note section DAA12

> Divide and conquer solves smaller largely independent instances and combines results. DP reuses a defined set of subproblem states. Greedy commits to a feasible local choice only when a safety argument supports it. Backtracking grows/retracts candidates when constraints fail; branch and bound also rejects candidates whose best possible objective cannot beat an incumbent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

> Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Design and Analysis of Algorithms module | PDF page 17 | lines 5-9

> 2.1 Divide and Conquer Algorithm In this approach, we solve a problem recursively by applying 3 steps 1. DIVIDE-break the problem into several sub problems of smaller size. 2. CONQUER-solve the problem recursively. 3. COMBINE-combine these solutions to create a solution to the original problem.

Question sources

Addis Ababa University mock exam · Question 79

Computer Science Exit Exam · Sene 2016 · Question 18-3

Computer Science Exit Exam practice collection · Question 117

Question 7 | Q00283

Suppose we want to determine the efficiency of the algorithm, then how we can measure the space factor

A. To count the maximum memory required by the algorithm

B. To count the average memory required by the algorithm

C. To count the minimum memory required by the algorithm

D. To count the maximum disk space needed by the algorithm

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: To count the maximum memory required by the algorithm

Space needed to run to completion must allow the program to finish without exhausting memory (PDF p.6). Hence the required allocation is peak simultaneously needed memory for that execution; average or minimum use may underallocate.

Why each choice is right or wrong

A - Correct: Maximum simultaneously required memory supports completion (p.6).

B - Incorrect: Average use can fall below a later peak and cannot guarantee enough memory (p.6).

C - Incorrect: Minimum use cannot supply the full execution requirement (p.6).

D - Incorrect: Disk capacity alone does not measure the program memory requirement defined on p.6.

Primary note proof

Study-note section DAA08

> Space complexity measures peak simultaneously live storage, not the sum of every allocation ever made. Count input, auxiliary structures and stack according to the requested convention. n active recursion levels with constant-size frames need Θ(n) stack; depth log n with constant frames needs Θ(log n). The number of total calls does not determine peak depth.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

> The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Question sources

Computer Science model exam · Question 14

Computer Science model exam 2 · Question 67

Computer Science model exam 2 · Question 89

University of Gondar model exam · Question 14

Question 8 | Q00284

Two main measures for the efficiency of an algorithm are

A. Processor and memory

B. Data and space

C. Complexity and capacity

D. Time and space

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: Time and space

Performance is stated in computer memory and time (PDF p.4), and complexity is expressed as space and time (p.6). The pair is therefore time and space.

Why each choice is right or wrong

A - Incorrect: Processor is a device, not the time measure paired with memory (p.4).

B - Incorrect: Data is the problem content, not one of the two resource measures (p.6).

C - Incorrect: Complexity and capacity are not the stated two measures (p.6).

D - Correct: Time and space are explicitly named on p.6.

Primary note proof

Study-note section DAA02

> Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

> 1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

> The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Question sources

Computer Science model exam · Question 15

University of Gondar model exam · Question 15

Question 9 | Q00285

For the standard Floyd-Warshall implementation on an n-vertex weighted graph, using constant-time distance arithmetic and the full k,i,j loops, what is the tight running-time bound?

A. $\Theta(n^2)$

B. $\Theta(n^3)$

C. $\Theta(n)$

D. $\Theta(n^4)$

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: $\Theta(n^3)$

Floyd-Warshall considers each intermediate vertex k and updates every ordered pair i,j. The three independent n-sized loop ranges perform $\Theta(n^3)$ constant-cost updates. The n^2 output table explains storage, not the time of this implementation. Other graph assumptions can support different algorithms, so this is the cost of the explicitly named method rather than a universal best-achievable APSP bound.

Why each choice is right or wrong

A - Incorrect: Quadratic counts pair storage but misses the intermediate-vertex loop.

B - Correct: Three full n-sized loops give cubic time.

C - Incorrect: Linear is too small even for the explicit pair table.

D - Incorrect: Quartic is a loose upper bound rather than the tight cost.

Primary note proof

Study-note section DAA28

> For all-pairs distances initialize D[i, i]=0, D[i, j] to the minimum direct edge weight, and missing edges to∞. Preserve a negative self-loop by taking min(0, its weight). For k outermost, update D[i, j]=min(D[i, j], D[i, k]+D[k, j]); before phase k, intermediates are restricted to earlier vertices. Guard unreachable sums and numeric overflow.

Study-note section DAA28

> Three loops give $\Theta(V^3)$ time and $\Theta(V^2)$ storage. A negative diagonal afterward identifies a negative cycle; pairs able to traverse it have no finite minimum walk distance. Warshall reachability replaces min/+ by OR/AND. Floyd's cost is not a universal best-achievable bound for all APSP settings; sparse nonnegative graphs permit repeated Dijkstra and other models permit other methods.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 61 | lines 14-23

> 4.2. ALL PAIR SHORTEST PATH • Let G=<V,E> be a directed graph 'V' is a set of nodes and _E' is the set of edges. • Each edge has an associated non-negative length. • We want to calculate the length of the shortest path between each pair of nodes. • Suppose the nodes of G are numbered from 1 to n, so V={1,2,...N},and suppose G matrix L gives the length of each edge, with L(i,j)=0 for i=1,2...n,L(i,j)>=for all i & j, and L(i,j)=infinity, if the edge (i,j) does not exist. • The principle of optimality applies: if k is the node on the shortest path from i to j then the part of the path from i to k and the part from k to j must also be optimal, that is shorter.

Design and Analysis of Algorithms module | PDF page 64 | lines 18-28

> Function AllPair (L[1..r,1..r]):array[1..n,1..n] array D[1..n,1..n] D = L For k = 1 to n do For i = 1 to n do For j = 1 to n do D [i , j] = min (D[i, j], D[i, k] + D[k, j] Return D ANALYSIS: This algorithm takes a time of O(n3)

Question sources

Computer Science model exam · Question 16

University of Gondar model exam · Question 16

Question 10 | Q00287

A recursive execution has n simultaneously active call levels, each retaining Theta(1) storage. Count only live call-frame storage. What is the tightest listed asymptotic upper bound on its peak space?

A. O(n'2)

B. O(n)

C. O(1)

D. O(log n)

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: B. O(n)

All n call frames are live at the deepest point. Each frame retains a fixed positive amount of storage, so their combined peak space is proportional to n. This gives Theta(n) space and makes O(n) the tightest listed upper bound. $O(n^2)$ is a looser valid upper bound, while constant and logarithmic space cannot hold for n simultaneously live constant-size frames.

Why each choice is right or wrong

A - Incorrect: Quadratic is a valid but looser upper bound than the requested tightest one.

B - Correct: Linear space tightly bounds n live constant-size frames.

C - Incorrect: Constant space cannot hold as the number of live frames grows.

D - Incorrect: Logarithmic space cannot hold for n live constant-size frames.

Primary note proof

Study-note section DAA08

> Space complexity measures peak simultaneously live storage, not the sum of every allocation ever made. Count input, auxiliary structures and stack according to the requested convention. n active recursion levels with constant-size frames need Θ(n) stack; depth log n with constant frames needs Θ(log n). The number of total calls does not determine peak depth.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 6 | lines 28-31

> • Environmental stack space:- is the space used to save information needed to resume execution of partially completed methods (functions). For example, if method func1 invokes or calls method func2, then we must at least save a pointer to the instruction of func1 to be executed when func2 terminates. Each time a function or method is called or invoked, the following data are stored on the

Design and Analysis of Algorithms module | PDF page 8 | lines 2-3

> ii. The values of all local variables and formal parameters in the method or function being invoked or called. Time Complexity of a program: - is the amount of computer time it needs to run to completion. We are

Data Structure and Algorithms module | PDF page 3 | lines 2-5

> I. Big-Oh Notation (O): It’s only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Question sources

Computer Science model exam · Question 18

University of Gondar model exam · Question 18

Question 11 | Q00295

Which of the following is a Divide and Conquer algorithm?

A. Merge Sort

B. Heap Sort

C. Selection Sort

D. Bubble Sort

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: Merge sort

DAA14 splits into halves, recursively sorts each, and merges the results, directly implementing divide and conquer. DAA16 excludes standard heap sort from that category and describes selection's repeated minimum and bubble's adjacent passes. Only Merge Sort among these options has the specified independent recursive decomposition and combination mechanism.

Why each choice is right or wrong

A - Correct: Merge sort recursively solves halves and combines them through a merge.

B - Incorrect: Standard heapsort builds and repeatedly restores a heap rather than applying the standard independent-subproblem decomposition.

C - Incorrect: Selection repeatedly finds a suffix minimum instead of recursively sorting halves.

D - Incorrect: Bubble repeatedly swaps adjacent inversions instead of recursively decomposing subinstances.

Primary note proof

Study-note section DAA14

> Split by indices, recursively sort each half, then repeatedly copy the smaller remaining front element into a temporary result. The merge invariant says the output prefix is sorted and contains the smallest elements already consumed. Append the unexhausted suffix. Equal keys must be taken from the left half first to preserve stability.

Study-note section DAA16

> Insertion shifts larger sorted-prefix keys: stable with strict comparisons, $\Theta(n)$ best and $\Theta(n^2)$ worst. Optimized bubble sort swaps adjacent inversions and stops after a no-swap pass: stable, $\Theta(n)$ best, $\Theta(n^2)$ worst. Selection repeatedly finds a suffix minimum: $\Theta(n^2)$ comparisons in every order, normally unstable. These use $\Theta(1)$ auxiliary array space.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 17 | lines 23-29

> • Breaking large problems into smaller subproblems • Example of algorithms uses divide-and-conquer strategy : • Binary search, • Finding maximum and minimum • Merge sort • Quick sort • selection sort

Design and Analysis of Algorithms module | PDF page 21 | lines 18-29

> MERGE SORT • As another example divide-and-conquer, we investigate a sorting algorithm that has the nice property that is the worst case its complexity is O(n log n) • This algorithm is called

merge sort • We assume throughout that the elements are to be sorted in non-decreasing order. • Given a sequence of _n' elements a[1],...,a[n] the general idea is to imagine then split into 2 sets a[1],.....,a[n/2] and a[[n/2]+1],....a[n]. • Each set is individually sorted, and the resulting sorted sequences are merged to produce a single sorted sequence of _n' elements. • Thus, we have another ideal example of the divide-and-conquer strategy in which the splitting is into 2 equal-sized sets & the combining operation is the merging of 2 sorted sets into one.

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items. • How many comparisons? => O(n2) and How many swaps? => O(n)

Question sources

Computer Science model exam · Question 26

University of Gondar model exam · Question 26

Question 12 | Q00373

In Average case, if operation takes ƒ(n) time in execution, then m operations will take?

A. ƒ(n)

B. f(m)

C. mf(n)

D. nf(m)

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C: mf(n)

Adding m execution costs each with mean f(n) gives mean total m*f(n) by the consecutive-block addition rule (PDF p.12), using the average-case meaning on p.14. This assumes n is fixed and each of the m operations has the stated mean; no independence is needed to add expected costs.

Why each choice is right or wrong

A - Incorrect: f(n) counts only one mean execution cost (pp.12,14).

B - Incorrect: f(m) changes the argument instead of adding m costs (p.12).

C - Correct: m equal mean costs sum to m*f(n) (p.12).

D - Incorrect: n*f(m) swaps operation count and input size without justification (p.12).

Primary note proof

Study-note section DAA06

If m operations each have expected cost f(n) at the same fixed size, expected total is m f(n), by linearity of expectation; independence is unnecessary. If sizes change, sum f(n_i). Exact coefficients depend on whether tests, increments, arithmetic and assignments count separately.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 12 | lines 6-20

for Loops: Formally • In general, a for loop translates to a summation. The index and bounds of the summation are the same as the index and bounds of the for loop. ∑1 = N i=1 • Suppose we count the number of additions that are done. There is 1 addition per iteration of the loop, hence N additions in total. Nested Loops: Formally • Nested for loops translate into multiple summations, one for each for loop. N M N ∑∑2 = ∑2M = 2MN i=1 j=1 i=1 • Again, count the number of additions. The outer summation is for the outer for loop. Consecutive Statements: Formally • Add the running times of the separate blocks of your code

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Question sources

Data Structures & Algorithms practice questions · Question 3

Question 13 | Q00376

O(1) means computing time is _____

A. Constant

B. Quadratic

C. Linear

D. Cubic

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: Constant

O(1) is a bounded constant upper growth rate independent of input size. The related PDF p.3 visually orders constant K before log n, n, n log n and n^2; the assigned PDF p.14 explains comparison by growth rate.

Why each choice is right or wrong

A - Correct: A constant bound does not increase with n (cross-module p.3 growth diagram).

B - Incorrect: Quadratic corresponds to n squared, which grows with n (assigned p.14).

C - Incorrect: Linear is proportional to n, unlike 1 (assigned p.14).

D - Incorrect: Cubic is proportional to n cubed, unlike 1 (assigned p.14).

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, $f \in O(g)$ means $f(n) \le c \cdot g(n)$ for all $n \ge n_0$, with fixed $c > 0$. Ω reverses that inequality; Θ requires both bounds. Little-o means $f/g \to 0$, and little-ω means $f/g \to \infty$ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Study-note section DAA05

> Representative growth is $1, \log n, n, n \log n, n^2, n^3, \ldots, b^n, n!$ for fixed $b > 1$. Log bases differ by a constant. Both $n \log n$ and n^2 are superlinear; $\log n$ is sublinear. Polynomial growth has a fixed exponent in input size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

> I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Question sources

Data Structures & Algorithms practice questions · Question 6

Question 14 | Q00378

Which of the following sorting algorithm is of divide-and-conquer type?

A. Bubble sort

B. Insertion sort

C. Quick sort

D. Selection sort

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C: Quicksort

DAA15 partitions around a pivot and recursively sorts smaller ranges, which is the divide-and-conquer strategy. The other ordinary methods in DAA16 use adjacent passes, sorted-prefix insertion, or repeated suffix-minimum selection. They do not use the same recursive partition decomposition. Thus Quick sort is the unique matching main choice.

Why each choice is right or wrong

A - Incorrect: Bubble uses repeated adjacent comparisons rather than recursive partitioning.

B - Incorrect: Insertion grows a sorted prefix by shifting larger preceding values.

C - Correct: Quicksort partitions and recursively sorts the resulting smaller ranges.

D - Incorrect: Selection repeatedly picks a suffix minimum rather than recursively partitioning and combining.

Primary note proof

Study-note section DAA15

> Choose a pivot, partition, then sort the resulting smaller ranges. In a two-pointer scheme, advance i while values are below pivot and j while above; when i≤j, swap and advance both. Bound the scans and ensure progress on equal values. Recurse on the resulting ranges. Hoare-style partition boundaries need not put the pivot at its final position; Lomuto-style partition commonly does. Do not mix their return conventions.

Study-note section DAA16

> Insertion shifts larger sorted-prefix keys: stable with strict comparisons, $\Theta(n)$ best and $\Theta(n^2)$ worst. Optimized bubble sort swaps adjacent inversions and stops after a no-swap pass: stable, $\Theta(n)$ best, $\Theta(n^2)$ worst. Selection repeatedly finds a suffix minimum: $\Theta(n^2)$ comparisons in every order, normally unstable. These use $\Theta(1)$ auxiliary array space.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 23 | lines 19-29

QUICK SORT • The divide-and-conquer approach can be used to arrive at an efficient sorting method different from merge sort. • In merge sort, the file a[1:n] was divided at its midpoint into sub arrays which were independently sorted & later merged. • In Quick sort, the division into 2 sub arrays is made so that the sorted sub arrays do not need to be merged later. • This is accomplished by rearranging the elements in a[1:n] such that a[I]<=a[j] for all I between 1 & n and all j between (m+1) & n for some m, 1<=m<=n. • Thus the elements in a[1:m] & a[m+1:n] can be independently sorted. • No merge is needed. This rearranging is referred to as partitioning.

Design and Analysis of Algorithms module | PDF page 21 | lines 18-29

MERGE SORT • As another example divide-and-conquer, we investigate a sorting algorithm that has the nice property that is the worst case its complexity is O(n log n) • This algorithm is called merge sort • We assume throughout that the elements are to be sorted in non-decreasing order. • Given a sequence of _n' elements a[1],...,a[n] the general idea is to imagine then split into 2 sets a[1],.....,a[n/2] and a[[n/2]+1],....a[n]. • Each set is individually sorted, and the resulting sorted sequences are merged to produce a single sorted sequence of _n' elements. • Thus, we have another ideal example of the divide-and-conquer strategy in which the splitting is into 2 equal-sized sets & the combining operation is the merging of 2 sorted sets into one.

Data Structure and Algorithms module | PDF page 3 | lines 44-48

B. Selection Sort Loop through the list/array from i=0 to n-1 and select the smallest element in the list and swap this value with value at position i. Repeat this step for the rest of the items. • How many comparisons? => O(n2) and How many swaps? => O(n)

Question sources

Data Structures & Algorithms practice questions · Question 8

Question 15 | Q00718

of an algorithm refers to defining the mathematical bound/framing of its run-time performance?

A. Space complexity

B. Asymptotic analysis

C. Time complexity

D. none of the above

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: Asymptotic analysis

Asymptotic analysis studies the limiting growth of running time with input size and groups functions by growth bounds (PDF p.14). Time complexity is the resource function being studied; the bounding analysis is asymptotic analysis.

Why each choice is right or wrong

A - Incorrect: Space concerns memory, not bounding the stated runtime (p.14).

B - Correct: The growth-rate and bounding analysis is explicitly titled Asymptotic Analysis (p.14).

C - Incorrect: Time complexity describes runtime; asymptotic analysis classifies its growth as asked (p.14).

D - Incorrect: The bounding concept is provided by the p.14 section, so “none” fails.

Primary note proof

Study-note section DAA02

> Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Study-note section DAA04

> For eventually nonnegative f, g, $f \in O(g)$ means $f(n) \le c \cdot g(n)$ for all $n \ge n_0$, with fixed $c > 0$. Ω reverses that inequality; Θ requires both bounds. Little-o means $f/g \to 0$, and little-ω means $f/g \to \infty$ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Question sources

Computer Science model exam 2 · Question 59

Question 16 | Q00719

Which described implementation uses dynamic programming through stored dependent subproblem values?

A. Multistage DAG shortest paths using stored reverse-topological cost-to-go values

B. Single-source shortest paths by Dijkstra on nonnegative edges

C. Merge sort

D. Chess game-tree search without cached subproblem values

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: Multistage DAG shortest paths using stored reverse-topological cost-to-go values

Multistage DAG paths can be solved by storing the best cost-to-go for each vertex and evaluating vertices in reverse topological order. Each value combines outgoing-edge costs with already solved successor values, which is dynamic programming. The specified Dijkstra implementation uses greedy finalization, merge sort uses independent halves, and uncached chess tree search does not reuse stored subproblem values as described.

Why each choice is right or wrong

A - Correct: The reverse-topological cost recurrence reuses solved successor values.

B - Incorrect: Dijkstra uses greedy minimum-distance finalization under nonnegative weights.

C - Incorrect: Merge sort divides into independent halves and merges them.

D - Incorrect: The stipulated uncached search does not reuse solved state values.

Primary note proof

Study-note section DAA12

> DP reuses a defined set of subproblem states. Greedy commits to a feasible local choice only when a safety argument supports it.

Study-note section DAA27

> For a DAG or multistage graph, compute cost-to-go in reverse topological order: F(t)=0 and F(u)=min over outgoing(u, v) of[w(u, v)+F(v)].

Study-note section DAA24

> Repeatedly extract the unsettled vertex u of minimum distance; relax each edge by dist[v]=min(dist[v], dist[u]+w(u, v)), updating predecessors when improved.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 49 | lines 2-12

The Principle of Optimality The dynamic programming relies on a principle of optimality. This principle states that in an optimal sequence of decisions or choices, each subsequence must also be optimal. For example, in matrix chain multiplication problem, not only the value we are interested in is optimal but all the other entries in the table are also represent optimal. The principle can be related as follows: the optimal solution to a problem is a combination of optimal solutions to some of its sub problems. The difficulty in turning the principle of optimally into an algorithm is that it is not usually obvious which sub problems are relevant to the problem under consideration. General method-multistage graphs-all pair shortest path algorithm-Optimal Binary Search trees- 0/1 knapsack Problem-Reliability design-traveling salesman problem

Design and Analysis of Algorithms module | PDF page 46 | lines 2-4

o Single Source Shortest Path Algorithm (Dijkstra's) Single-source shortest path: Graphs can be used to represent the highway structure of a state or country with

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Question sources

Computer Science model exam 2 · Question 60

Question 17 | Q00720

Optimal merge pattern is an example of _____ Algorithm design Approach

A. Dynamic Programming

B. Divide and Conquered

C. Greedy

D. Back tracking

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C: Greedy

The optimal-merge construction explicitly chooses the two smallest files at each step (PDF p.43). That repeated local selection is called a greedy attempt on the page and gives the listed example an optimal merge pattern.

Why each choice is right or wrong

A - Incorrect: No table of overlapping subproblems drives the p.43 construction.

B - Incorrect: The pairing order is chosen by repeated smallest-file selection, not a predetermined independent split (p.43).

C - Correct: The module explicitly calls this selection rule greedy (p.43).

D - Incorrect: The described rule does not backtrack over earlier pairings (p.43).

Primary note proof

Study-note section DAA22

> Combine the two smallest weights, add their sum to total cost, and reinsert it. A min-heap gives O(n log n) time/O(n) space. Cost p+q counts moved records; worst-case comparisons are p+q−1 for nonempty lists, and actual comparisons may be fewer.

Study-note section DAA19

> A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it. Optimal substructure alone does not prove this safe. An exchange proof inserts the greedy choice into an optimum without worsening cost; recurse. MST cut proofs use this exchange.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 43 | lines 5-12

> A greedy attempt to obtain an optimal merge pattern is easy to formulate. Since merging an n record file and an m record file requires possibly n + m records moves, the obvious choice for a selection criterion is: at each step merge the two smallest size files together. Thus, if we have five files F , ••• , F 5) with sizes (20, 30, 10, 5, 30) our greedy rule would generate the following merge pattern: merge F4 and FJ to get Z1 (|Z1| = 15); merge Z1 and F1 to get Z2 (|Z2| = 35); merge Fi and Fs to get ZJ (|ZJ| = 60); merge Z2 and ZJ to get the answer Z4. The total number of record moves is 205. One can verify that this is an optimal merge pattern for the given problem instance.

Question sources

Computer Science model exam 2 · Question 61

Question 18 | Q00898

For each input length n, a report records the smallest number of comparisons attained over all legal inputs of that length. Which case-analysis function is the report recording?

A. Average time

B. Small time

C. .

Worst time

D. Best time

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: D. Best time

At each fixed length n, the report selects the legal input with the smallest comparison count. This records best-case time under a comparison-count model. Worst case would record the largest count, and average case would take an expectation under a specified distribution. Comparing different input lengths is a separate growth analysis, so the application preserves the fixed-size meaning of Best time.

Why each choice is right or wrong

A - Incorrect: Average time is an expectation, not the minimum.

B - Incorrect: Small time is not the standard case-analysis term.

C - Incorrect: Worst time is the maximum over inputs of the fixed size.

D - Correct: Best time is the minimum over legal inputs of that fixed size.

Primary note proof

Study-note section DAA03

> At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

> In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the

smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Question sources

Computer Science Exit Exam · 2018 · Question 11

Question 19 | Q00925

Which statement about proving worst-case lower bounds for all comparison-based sorting algorithms is FALSE?

A. A comparison decision tree must distinguish n! possible orders of distinct keys.

B. Solving a recurrence for one chosen sorting algorithm, by itself, proves a lower bound for every algorithm in the comparison class.

C. Suppressing constant factors in asymptotic notation does not by itself invalidate a lower-bound proof.

D. A decision-tree argument can establish an Ω(n log n) worst-case comparison bound even when array indexing is allowed.

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: B. A single algorithm’s recurrence alone does not prove a class-wide lower bound.

A recurrence describes the running time of the particular algorithm it models. An upper or tight bound for that algorithm does not exclude a different comparison algorithm using fewer operations, so B is false. The class-wide decision-tree argument instead distinguishes n! orders of distinct keys and needs height at least $\log_2(n!)$, giving Ω(n log n). Ignoring constants and allowing ordinary array access do not defeat that comparison-count argument.

Why each choice is right or wrong

A - Incorrect: This is a true premise of the comparison decision-tree lower-bound argument, so it is not the requested false statement.

B - Correct: This is false: a recurrence for one algorithm does not lower-bound every other algorithm.

C - Incorrect: This is true: asymptotic lower bounds intentionally disregard constant factors.

D - Incorrect: This is true: the bound concerns comparisons, not a prohibition on array indexing.

Primary note proof

Study-note section DAA11

A comparison decision tree needs at least n! leaves for distinct-key orderings. Height h allows at most 2^h leaves, so h≥log 2(n!)=Ω(n log n); merge sort matches this worst-case bound. Array indexing is not forbidden.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 21 | lines 18-29

> MERGE SORT • As another example divide-and-conquer, we investigate a sorting algorithm that has the nice property that is the worst case its complexity is O(n log n) • This algorithm is called merge sort • We assume throughout that the elements are to be sorted in non-decreasing order. • Given a sequence of _n' elements a[1],...,a[n] the general idea is to imagine then split into 2 sets a[1],.....,a[n/2] and a[[n/2]+1],....a[n]. • Each set is individually sorted, and the resulting sorted sequences are merged to produce a single sorted sequence of _n' elements. • Thus, we have another ideal example of the divide-and-conquer strategy in which the splitting is into 2 equal-sized sets & the combining operation is the merging of 2 sorted sets into one.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Design and Analysis of Algorithms module | PDF page 2 | lines 9-11

> - Describe Θ(f), Ω(f), O(f), growth rate, and algorithm order. - Use decision tree to determine a lower bound on complexity - Convert a simple recurrence relation into its closed form.

Question sources

Computer Science Exit Exam · 2018 · Question 38

Question 20 | Q00929

For integer n≥0, treat the code below as mathematical-integer RAM pseudocode: arithmetic is exact and unit-cost, with no overflow. Charge 1 unit for each initialization, comparison, loop increment and return; charge 1 for the addition and 1 for assignment in sum=sum+1. Ignore declarations, parameter passing and function-call overhead. What is T(n)?
total(n) { sum=0; for(i=1;i<=n;i++) sum=sum+1; return sum; }

A. $5n+5$

B. $4n^2+4$

C. $6n+2$

D. 4n+4

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: 4n+4

In this exact-integer model, the two initializations cost 2 units. The test i<=n runs n+1 times, the increment runs n times, and addition plus assignment costs 2n units. The return costs one unit. Thus T(n)=2+(n+1)+n+2n+1=4n+4, including n=0. Unbounded mathematical arithmetic keeps the final increment defined throughout the stated domain.

Why each choice is right or wrong

A - Incorrect: 5n+5 does not match the specified operation counts.

B - Incorrect: Quadratic growth does not match a single n-iteration constant-body loop.

C - Incorrect: 6n+2 charges operations differently from the stated model.

D - Correct: The complete count is 2+(n+1)+n+2n+1=4n+4.

Primary note proof

Study-note section DAA06

> If m operations each have expected cost f(n) at the same fixed size, expected total is m f(n), by linearity of expectation; independence is unnecessary. If sizes change, sum f(n_i). Exact coefficients depend on whether tests, increments, arithmetic and assignments count separately.

Study-note section DAA06

> For sum=0; for(i=1;i<=n;i++) sum=sum+1; return sum: counting two initializations, n+1 tests, n increments,2n body operations and one return gives 4n+4. A different primitive-cost convention changes this exact expression, not Θ(n).

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 10 | lines 6-20

> Analysis Rules: 1. We assume an arbitrary time unit. 2. Execution of one of the following operations takes time 1: • Assignment Operation • Single Input/output Operation • Single Boolean Operations • Single Arithmetic Operations • Function Return 3. Running time of a selection statement (if, switch) is the time for the condition evaluation plus the maximum of the running times for the individual clauses in the selection. 4. Loops: Running time for a loop is equal to the running time for the statements inside the loop multiplied by the number of iterations. The total running time of a statement inside a group of nested loops is the running time of the statements multiplied by the product of the sizes of all the loops. For nested loops, analyze inside out.

Design and Analysis of Algorithms module | PDF page 10 | lines 34-41

1 for the assignment statement: int k=0 1 for the output statement. 1 for the input statement. In the for loop: 1 assignment, n+1 tests, and n increments. n loops of 2 units for an assignment, and an addition. 1 for the return statement. T (n)= 1+1+1+(1+n+1+n)+2n+1 = 4n+6 = O(n)

Question sources

Computer Science Exit Exam · 2018 · Question 42

Question 21 | Q01211

Which option most completely covers the listed factors that affect a program's exact elapsed running time, including its workload and execution environment?

A. Processor speed

B. Current processor load

C. Input size of the given algorithm

D. All

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: All listed factors affect observed runtime.

Input size influences how many operations the algorithm performs. Processor speed influences how quickly that work executes, and competing processor load influences the resources available during the run. All three can therefore affect exact elapsed time. Each individual factor is true but incomplete as an answer to the combined-factor question; D includes the whole set without confusing elapsed measurements with an abstract operation-count bound.

Why each choice is right or wrong

A - Incorrect: True but incomplete: processor speed matters, but it omits current load and input size.

B - Incorrect: True but incomplete: competing load matters, but it omits processor speed and input size.

C - Incorrect: True but incomplete: input size matters, but it omits the execution-environment factors.

D - Correct: All three listed factors contribute to exact elapsed time.

Primary note proof

Study-note section DAA02

Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs

need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Design and Analysis of Algorithms module | PDF page 6 | lines 18-30

Components of space complexity: - the space needed by a program has the following components: • Instruction space: - is the amount of space needed to store the compiled version of the program instructions. The amount of instruction space that is needed depends on factors such as: - The compiler used to compile the program into machine code. - The compiler options in effect at time of compilation. - The target computer. • Data space: - is the space needed to store all constant and variable values. Data space has two components: i. Space needed by constants (e.g. the numbers 0, 1, 2...) and simple variables. ii. Space needed by dynamically allocated objects such as arrays and class instances. • Environmental stack space:- is the space used to save information needed to resume execution of partially completed methods (functions). For

example, if method func1 invokes or calls method func2, then we must at least save a pointer to the instruction of func1 to be executed when func2

Question sources

Computer Science model exam 2 · Question 32

Question 22 | Q01230

Which of the following is an example of a Dynamic Programming algorithm?

A. Breadth First Search

B. Depth First Search

C. Dijkstra's Algorithm

D. Knapsack Problem

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: Knapsack Problem

The module develops 0/1 knapsack in the dynamic-programming unit (PDF p.71; application list p.49), with alternatives of including or excluding an item. Dijkstra is placed in the greedy unit (p.46). BFS and DFS are traversal techniques, rather than the shared-subproblem optimization example intended here (p.85).

Why each choice is right or wrong

A - Incorrect: Breadth-first search is a traversal; p.85 separates graph search from the knapsack optimization example.

B - Incorrect: Depth-first search is a traversal; the dynamic-programming item-selection example is p.71.

C - Incorrect: Dijkstra appears under the greedy unit on p.46, not as the intended dynamic-programming example.

D - Correct: The 0/1 knapsack variant is explicitly developed in the dynamic-programming unit on p.71.

Primary note proof

Study-note section DAA29

For integer capacity W, K[i, w] is maximum value using the first i items within capacity w. Initialize K[0, w]=0. If weight_i>w, use K[i−1, w]; otherwise take max(K[i−1, w], value_i+K[i−1, w−weight_i]). The previous-row reference ensures each item is used at most once. Backtrack changed values to recover chosen items.

Study-note section DAA34

BFS/DFS being printed under a DP unit does not make ordinary traversal a DP optimization algorithm.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Design and Analysis of Algorithms module | PDF page 71 | lines 3-9

4.4. 0/1 KNAPSACK PROBLEM • This problem is similar to ordinary knapsack problem but we may not take a fraction of an object. • We are given _ N _ objects with weight Wi and profits Pi where i varies from l to N and also a knapsack with capacity _ M _. • The problem is, we have to fill the bag with the help of _ N _ objects and the resulting profit has to be maximum.

Design and Analysis of Algorithms module | PDF page 46 | lines 2-4

o Single Source Shortest Path Algorithm (Dijkstra’s) Single-source shortest path: Graphs can be used to represent the highway structure of a state or country with

Design and Analysis of Algorithms module | PDF page 85 | lines 14-20

4.7. BASIC SEARCH AND TRAVERSAL TECHNIQUE DEFINING GRAPH: A graphs g consists of a set V of vertices (nodes) and a set E of edges (arcs). We write G=(V,E). V is a finite and non-empty set of vertices. E is a set of pair of vertices; these pairs are called as edges. Therefore, V(G) is read as V of G, is a set of vertices and E(G),read as E of G is a set of edges. An edge e= (v, w) is a pair of vertices v and w, and to be incident with v and w.

Question sources

Computer Science model exam 2 · Question 51

Computer Science model exam 2 · Question 58

Question 23 | Q01232

When designing an algorithm for a specified implementation environment, which option most completely covers the listed resource, correctness and alternative-method considerations?

A. Whether the method respects the chosen hardware/resource cost model.

B. Whether the software implementation correctly realizes the specified method.

C. If there is more than one way to solve the problem.

D. All of the above

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: All of the above

Algorithm design starts with inputs, outputs, constraints and correctness, then evaluates resource use under a stated model. For an implementation, respecting the selected hardware/resource assumptions and ensuring the program implements the method correctly both matter. Comparing alternative solution methods is also legitimate. Thus the all-of-the-above option combines all three considerations; none of the individual choices alone covers the entire set.

Why each choice is right or wrong

A - Incorrect: Respecting the resource model matters but is only one consideration.

B - Incorrect: Implementation correctness matters but is only one consideration.

C - Incorrect: Comparing alternatives matters but is only one consideration.

D - Correct: All three stated considerations belong in the implementation-aware design.

Primary note proof

Study-note section DAA01

> State inputs, output, constraints and empty-solution policy. Algorithms are finite, definite executable methods independent of source language; feasible satisfies constraints, optimal is best feasible. Conditional branches are legitimate control flow.

Study-note section DAA02

> Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 6-19

• Finiteness: Algorithm must complete after a finite number of steps. • Definiteness: Each step must be clearly defined, having one and only one interpretation. At each point in computation, one should be able to tell exactly what happens next. • Sequence: Each step must have a unique defined preceding and succeeding step. The first step (start step) and last step (halt step) must be clearly noted. • Feasibility: It must be possible to perform each instruction. • Correctness: It must compute correct answer for all possible legal inputs. • Language Independence: It must not depend on any one programming language. • Completeness: It must solve the problem completely. • Effectiveness: It must be possible to perform each step exactly and in a finite amount of time. • Efficiency: It must solve with the least amount of computational resources such as time and space. • Generality: Algorithm should be valid on all possible inputs • Input/Output: There must be a specified number of input values, and one or more result values.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Design and Analysis of Algorithms module | PDF page 6 | lines 18-30

Components of space complexity: - the space needed by a program has the following components: • Instruction space: - is the amount of space needed to store the compiled version of the program instructions. The amount of instruction space that is needed depends on factors such as: - The compiler used to compile the program into machine code. - The compiler options in effect at time of compilation. - The target computer. • Data space: - is the space needed to store all constant and variable values. Data space has two components: i. Space needed by constants (e.g. the numbers 0, 1, 2...) and simple variables. ii. Space needed by dynamically allocated objects such as arrays and class instances. • Environmental stack space:- is the space used to save information needed to resume execution of partially completed methods (functions). For example, if method func1 invokes or calls method func2, then we must at least save a pointer to the instruction of func1 to be executed when func2

Question sources

Computer Science model exam 2 · Question 53

Question 24 | SM0007

Assume you have a weighted graph G with a set of vertices V and set of edges E. Which one of the following is an optimization problem on the graph?

A. Traverse all vertices in graph G

B. Is vertex p in V reachable from vertex q in V in graph G

C. Find minimum spanning tree from graph G

D. How many cycles are there in graph G

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C: Find minimum spanning tree from graph G

An optimization problem seeks a feasible solution minimizing or maximizing an objective (PDF p.27). The minimum spanning tree minimizes total edge weight while connecting every vertex (p.36); the other tasks ask for traversal, a yes/no fact or a count.

Why each choice is right or wrong

A - Incorrect: Traversal asks to visit vertices rather than optimize a weight (pp.27,36).

B - Incorrect: Reachability is a decision question rather than minimization (pp.27,85).

C - Correct: Minimum total tree weight is the optimization objective (p.36).

D - Incorrect: Counting cycles asks for a number, with no minimization or maximization criterion (p.27).

Primary note proof

Study-note section DAA01

> State inputs, output, constraints and empty-solution policy. Algorithms are finite, definite executable methods independent of source language; feasible satisfies constraints, optimal is best feasible. Conditional branches are legitimate control flow.

Study-note section DAA23

> For a connected weighted undirected graph, an MST spans every vertex without cycles and minimizes total edge weight; it has V−1 edges. Negative edges are allowed. Equal weights may yield multiple MSTs; distinct weights suffice for uniqueness. A tree is itself a graph.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 27 | lines 7-18

> • Greedy method is the most straightforward designed technique. • As the name suggest they are short sighted in their approach taking decision on the basis of the information immediately at the hand without worrying about the effect these decisions may have in the future. DEFINITION:

• A problem with N inputs will have some constraints. Any subsets that satisfy these constraints are called a feasible solution. • A feasible solution that either maximize can minimize a given objectives function is called an optimal solution. Greedy is a strategy that works well on optimization problems with the following characteristics: • Greedy-choice property: A global optimum can be arrived at by selecting a local

Design and Analysis of Algorithms module | PDF page 36 | lines 16-33

o Minimum Cost Spanning Tree- Prim's Algorithm ✓ Let G(V,E) be an undirected connected graph with vertices v and edge E . ✓ A sub-graph t=(V,E) of the G is a Spanning tree of G iff t is a tree.3 ✓ The problem is to generate a graph G = (V,E) where E is the subset of E,G is a Minimum spanning tree. ✓ Each and every edge will contain the given non-negative length. connect all the nodes with edge present in set E and weight has to be minimum. NOTE: ✓ We have to visit all the nodes. ✓ The subset tree (i.e) any connected graph with N vertices must have at least N-1 edges and also it does not form a cycle. Definition: 1. A spanning tree of a graph is an undirected tree consisting of only those edge that are necessary to connect all the vertices in the original graph. 2. A Spanning tree has a property that for any pair of vertices there exist only one path between them and the insertion of an edge to a spanning tree form a unique cycle.

Design and Analysis of Algorithms module | PDF page 85 | lines 14-20

4.7. BASIC SEARCH AND TRAVERSAL TECHNIQUE DEFINING GRAPH: A graphs g consists of a set V of vertices (nodes) and a set E of edges (arcs). We write G=(V,E). V is a finite and non-empty set of vertices. E is a set of pair of vertices; these pairs are called as edges. Therefore, V(G) is read as V of G, is a set of vertices and E(G),read as E of G is a set of edges. An edge e= (v, w) is a pair of vertices v and w, and to be incident with v and w.

Question sources

MOE Computer Science Exit Exam · Question 2-3

National Computer Science Exit Exam · Question 90

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 107

Question 25 | SM0031

For an average-cost function f(n)=E[T(n)] under a specified input distribution, which notation expresses a tight constant-factor growth bound relative to g(n)?

A. $f(n) \in \Omega(g(n))$

B. $f(n) \in O(g(n))$

C. $f(n) \in \omega(g(n))$

D. $f(n) \in \Theta(g(n))$

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: f(n)∈Θ(g(n))

Average case first takes an expectation under the specified distribution. A tight growth relation for that resulting function requires both an eventual upper and lower constant-factor bound relative to g, which is Θ. Big-O supplies only an upper bound, Ω only a lower bound, and little-ω a strict asymptotic lower relation. None of the symbols intrinsically means average case.

Why each choice is right or wrong

A - Incorrect: Ω supplies a lower bound without the matching upper bound.

B - Incorrect: O supplies an upper bound without the matching lower bound.

C - Incorrect: Little-ω means f/g tends to infinity rather than constant-factor equivalence.

D - Correct: Θ states both constant-factor upper and lower bounds.

Primary note proof

Study-note section DAA03

> At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Study-note section DAA04

> For eventually nonnegative f, g, f∈O(g) means f(n)≤c·g(n) for all n≥n 0, with fixed c>0. Ω reverses that inequality; Θ requires both bounds. Little-o means f/g→0, and little-ω means f/g→∞ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

> In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

> I. Big-Oh Notation (O): It’s only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An

upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Data Structure and Algorithms module | PDF page 3 | lines 6-12

II. Big-Omega Notation (Ω): provides an asymptotic lower bound; f(n)= Ω(g (n)) means that the growth rate of f(n) is greater than or equal to g(n). III. Theta Notation (Θ): If f(n)= Θ (g(n)), then g(n) is an asymptotically tight bound for f(n) i.e f(n) and g(n) have the same rate of growth. IV. Little-o Notation (o): f(n)=o(g(n)) means f(n) has less growth rate compared to g(n). V. Little-Omega Notation (ω): ω notation used to denote a lower bound that is not asymptotically tight.

Question sources

MOE Computer Science Exit Exam · Question 9-3

National Computer Science Exit Exam · Question 11

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 110

Question 26 | SM0065

Size of large data files can be reduced to save storage disk space. Which algorithm is appropriate to reduce the size of large files?

A. Heap sort algorithm

B. Prim's algorithm

C. Merge sort algorithm

D. Huffman encoding algorithm

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: D. Huffman encoding algorithm

Huffman coding assigns prefix-free bit strings to symbols using their frequencies, minimizing the weighted code length in its model. It can therefore reduce the storage required for data represented with those codes. Heap sort and merge sort instead reorder items, while Prim builds a minimum spanning tree of a weighted graph. Those other objectives do not encode file contents more compactly. Metadata overhead still prevents a universal compression guarantee for every file.

Why each choice is right or wrong

A - Incorrect: Heap sort reorders items by repeatedly extracting a heap extreme; it does not encode file symbols.

B - Incorrect: Prim minimizes graph-connection weight, not encoded file length.

C - Incorrect: Merge sort orders records by recursively sorting and merging, rather than encoding symbols.

D - Correct: Huffman builds a frequency-based prefix code minimizing its weighted length.

Primary note proof

Study-note section DAA22

> Huffman uses frequencies as weights and 0/1 branches to leaf symbols, making a prefix-free code. Cost is∑frequency×depth.

Study-note section DAA16

> Heap sort builds a max-heap in Θ(n), then extracts/restores in O(log n) per item: O(n log n) total, Θ(1) auxiliary space, ordinarily unstable.

Study-note section DAA23

> For a connected weighted undirected graph, an MST spans every vertex without cycles and minimizes total edge weight; it has V−1 edges.

Study-note section DAA14

> Split by indices, recursively sort each half, then repeatedly copy the smaller remaining front element into a temporary result.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 45 | lines 21-29

> The cost of decoding a code word is proportional to the number of bits in the code. This number is equal to the distance of the corresponding external node from the root node. If qi is the relative frequency with which message Mi will be transmitted, then the expected decode time is Σ ≤i≤n+ qidi. where di is the distance of the external node for message Mi from the root node. The expected decode time is minimized by choosing code words resulting in a decode tree with minimal weighted external path length! Note that Σ ≤i≤n+ qidi is also the expected length of a transmitted message. Hence the code which minimizes expected decode time also minimizes the expected length of a message.

Design and Analysis of Algorithms module | PDF page 37 | lines 4-11

> Minimum cost spanning tree: o The cost of a spanning tree is the sum of cost of the edges in that trees. o There are 2 method to determine a minimum cost spanning tree are • Kruskal s Algorithm • Prom s Algorithm. • PRIM'S ALGORITHM Start from an arbitrary vertex (root). At each stage, add a new branch (edge) to the tree already constructed; the algorithm halts when all the vertices in

Design and Analysis of Algorithms module | PDF page 21 | lines 18-29

> MERGE SORT • As another example divide-and-conquer, we investigate a sorting algorithm that has the nice property that is the worst case its complexity is O(n log n) • This algorithm is called

merge sort • We assume throughout that the elements are to be sorted in non-decreasing order. • Given a sequence of _n‘ elements a[1],…,a[n] the general idea is to imagine then split into 2 sets a[1],…..,a[n/2] and a[[n/2]+1],….a[n]. • Each set is individually sorted, and the resulting sorted sequences are merged to produce a single sorted sequence of _n‘ elements. • Thus, we have another ideal example of the divide-and-conquer strategy in which the splitting is into 2 equal-sized sets & the combining operation is the merging of 2 sorted sets into one.

Question sources

MOE Computer Science Exit Exam · Question 19-1

National Computer Science Exit Exam · Question 2

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 111

Question 27 | SM0083

For the worst-cost function f(n)=max over size-n inputs T(input), which notation expresses a tight constant-factor bound relative to g(n)?

A. $f(n) \in \omega(g(n))$

B. $f(n) \in \Theta(g(n))$

C. $f(n) \in o(g(n))$

D. $f(n) \in \Omega(g(n))$

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: $f(n) \in \Theta(g(n))$

Worst case maximizes cost over all inputs of a fixed size. Once that function is defined, Θ expresses matching constant-factor upper and lower bounds relative to g. Little-o and little-ω describe strict asymptotic relations, and Ω alone supplies only a lower bound. The answer concerns tightness of the bound, not a claim that any notation inherently selects the worst input case.

Why each choice is right or wrong

A - Incorrect: Little-ω gives a strict lower-growth relation.

B - Correct: Θ gives matching constant-factor upper and lower bounds.

C - Incorrect: Little-o gives a strict upper-growth relation.

D - Incorrect: Ω alone lacks the required upper bound.

Primary note proof

Study-note section DAA03

At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Study-note section DAA04

For eventually nonnegative f, g, f∈O(g) means f(n)≤c·g(n) for all n≥n 0, with fixed c>0. Ω reverses that inequality; Θ requires both bounds. Little-o means f/g→0, and little-ω means f/g→∞ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

I. Big-Oh Notation (O): It’s only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Data Structure and Algorithms module | PDF page 3 | lines 6-12

II. Big-Omega Notation (Ω): provides an asymptotic lower bound; f(n)= Ω(g (n)) means that the growth rate of f(n) is greater than or equal to g(n). III. Theta Notation (Θ): If f(n)= Θ (g(n)), then g(n) is an asymptotically tight bound for f(n) i.e f(n) and g(n) have the same rate of growth. IV. Little-o Notation (o): f(n)=o(g(n)) means f(n) has less growth rate compared to g(n). V. Little-Omega Notation (ω): ω notation used to denote a lower bound that is not asymptotically tight.

Question sources

MOE Computer Science Exit Exam · Question 23-4

National Computer Science Exit Exam · Question 71

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 112

Question 28 | SM0087

Which one of the following algorithms can be used to extract a Minimum Spanning Tree from an input graph?

A. Prim's algorithm

B. Huffman encoding algorithm

C. Dijkstra's algorithm

D. Merge sort algorithm

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: Prim's algorithm

The module explicitly names Prim among the two minimum-cost spanning-tree methods (PDF p.37). Huffman codes optimize expected message length (p.45), Dijkstra is single-source path search (p.46), and merge sort orders arrays (p.21).

Why each choice is right or wrong

A - Correct: Prim is explicitly a minimum-cost spanning-tree method (p.37).

B - Incorrect: Huffman builds codes, not a spanning tree of the input graph (p.45).

C - Incorrect: Dijkstra targets source-to-vertex paths, a different objective (p.46).

D - Incorrect: Merge sort combines sorted sequences, not graph edges (p.21).

Primary note proof

Study-note section DAA23

> For a connected weighted undirected graph, an MST spans every vertex without cycles and minimizes total edge weight; it has V−1 edges. Negative edges are allowed. Equal weights may yield multiple MSTs; distinct weights suffice for uniqueness. A tree is itself a graph.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 37 | lines 4-11

> Minimum cost spanning tree: o The cost of a spanning tree is the sum of cost of the edges in that trees. o There are 2 method to determine a minimum cost spanning tree are • Kruskal s Algorithm • Prom s Algorithm. • PRIM'S ALGORITHM Start from an arbitrary vertex (root). At each stage, add a new branch (edge) to the tree already constructed; the algorithm halts when all the vertices in

Design and Analysis of Algorithms module | PDF page 45 | lines 21-29

The cost of decoding a code word is proportional to the number of bits in the code. This number is equal to the distance of the corresponding external node from the root node. If qi is the relative frequency with which message Mi will be transmitted, then the expected decode time is Σ ≤i≤n+ qidi. where di is the distance of the external node for message Mi from the root node. The expected decode time is minimized by choosing code words resulting in a decode tree with minimal weighted external path length! Note that Σ ≤i≤n+ qidi is also the expected length of a transmitted message. Hence the code which minimizes expected decode time also minimizes the expected length of a message.

Design and Analysis of Algorithms module | PDF page 46 | lines 2-4

o Single Source Shortest Path Algorithm (Dijkstra's) Single-source shortest path: Graphs can be used to represent the highway structure of a state or country with

Design and Analysis of Algorithms module | PDF page 21 | lines 18-29

MERGE SORT • As another example divide-and-conquer, we investigate a sorting algorithm that has the nice property that is the worst case its complexity is O(n log n) • This algorithm is called merge sort • We assume throughout that the elements are to be sorted in non-decreasing order. • Given a sequence of _n' elements a[1],...,a[n] the general idea is to imagine then split into 2 sets a[1],.....,a[n/2] and a[[n/2]+1],....a[n]. • Each set is individually sorted, and the resulting sorted sequences are merged to produce a single sorted sequence of _n' elements. • Thus, we have another ideal example of the divide-and-conquer strategy in which the splitting is into 2 equal-sized sets & the combining operation is the merging of 2 sorted sets into one.

Question sources

MOE Computer Science Exit Exam · Question 24-4

National Computer Science Exit Exam · Question 19

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 113

Question 29 | SM0214

Which analysis case calculates the upper bound on running time of an algorithm?

A. Optimal case

B. Best case

C. Average case

D. Worst case

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: Worst case

Worst-case time is the maximum time over the inputs under consideration and therefore bounds all their runtimes (PDF p.14). This is the case-analysis upper bound, distinct from the separate question of an asymptotic bound on that maximum.

Why each choice is right or wrong

A - Incorrect: Optimal case is not a defined case function on p.14.

B - Incorrect: Best case minimizes rather than bounds all runtime values from above (p.14).

C - Incorrect: An average need not exceed every observation (p.14).

D - Correct: Worst case supplies the all-input bound explicitly described on p.14.

Primary note proof

Study-note section DAA03

> At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

> In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 5-1

Computer Science Exit Exam practice collection · Question 118

Question 30 | SM0249

An algorithm has unit-cost running-time function $T(n)=3n^k+7n+12$, where k is a fixed integer with $k\geq 2$ independent of n. Which listed growth class describes it?

A. Linear

B. Polynomial

C. Constant

D. Exponential

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: Polynomial

A fixed exponent $k \geq 2$ makes the leading term $3n^k$ dominate $7n+12$. The resulting function has $\Theta(n^k)$ growth, which is polynomial. It is not linear because k exceeds one, and the positive growing terms exclude constant growth. Exponential growth would instead put the input in an exponent of a fixed base, a different form from the fixed-degree polynomial here.

Why each choice is right or wrong

A - Incorrect: $k \geq 2$ prevents linear growth.

B - Correct: A fixed degree k gives polynomial growth.

C - Incorrect: The positive n-dependent terms prevent constant growth.

D - Incorrect: A fixed-degree power is not exponential growth.

Primary note proof

Study-note section DAA05

> Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n * log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 16-2

Question 31 | SM0302

The amount of memory needed to run an algorithm to completion is known as:

A. Space complexity

B. Time complexity

C. Worst case

D. Best case

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: Space complexity

Space complexity is explicitly defined as the memory needed to run to completion (PDF p.6). This is a resource quantity, not an input-case category.

Why each choice is right or wrong

A - Correct: This is the exact definition on p.6.

B - Incorrect: Time measures duration rather than memory (p.6 contrast).

C - Incorrect: Worst case selects inputs; it is not the memory-resource name (p.6).

D - Incorrect: Best case selects inputs; it is not the memory-resource name (p.6).

Primary note proof

Study-note section DAA08

> Space complexity measures peak simultaneously live storage, not the sum of every allocation ever made. Count input, auxiliary structures and stack according to the requested convention. n active recursion levels with constant-size frames need Θ(n) stack; depth log n with constant frames needs Θ(log n). The number of total calls does not determine peak depth.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

> The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 33-2

Computer Science Exit Exam practice collection · Question 120

Question 32 | SS116

Interpret the following as mathematical-integer pseudocode with constant-cost arithmetic and print operations. Which offered Big-O upper bounds on the full running time are valid? Select all that apply; the bounds need not be tight.
for i=0,...,n−1:
for j=i,...,i^2−1:
if j mod i == 0:
for k=0,...,j−1: print("*")
An upper limit below the lower limit makes an empty loop.

A. $O(n^5)$

B. $O(n^7)$

C. O(n log n)

D. $O(n^2)$

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A, B: $O(n^5)$; $O(n^7)$

For i=0 and i=1 the j loop is empty, so no division by zero occurs. For i≥2, successful j values are i,2i,...,(i−1)i. Their k-loop work totals $i^2(i-1)/2=\Theta(i^3)$, and summing over i yields $\Theta(n^4)$. Consequently both $O(n^5)$ and $O(n^7)$ are valid loose upper bounds, while the two smaller offered rates cannot bound the work.

Why each choice is right or wrong

A - Correct: A quartic function is also $O(n^5)$.

B - Correct: A quartic function is also $O(n^7)$.

C - Incorrect: n log n is too small to upper-bound quartic work.

D - Incorrect: n^2 is too small to upper-bound quartic work.

Primary note proof

Study-note section DAA07

> For the loop i=0.. n−1, j=i.. i^2−1, and k=0.. j−1 only when j%i==0, i=0 and 1 have empty j ranges, so no remainder by zero is evaluated. For i≥2, successful j values are i,2i,...,(i−1)i. The inner work is $i\sum m=1..\ i-1\ m=i^2(i-1)/2=\Theta(i^3)$. Summing over i gives $\Theta(n^4)$; condition checks add only $\Theta(n^3)$.

Study-note section DAA07

> Neither n^5 nor n^7 is tight for this loop, though both are valid Big-O upper bounds. This assumes mathematical integers and constant-cost output; literal signed C++ overflow in i*i is undefined,

and executable runtime analysis requires valid syntax in a specified language. Redeclaring parameter n and using return @;3 are syntax errors; neither establishes logarithmic complexity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 12 | lines 6-20

for Loops: Formally • In general, a for loop translates to a summation. The index and bounds of the summation are the same as the index and bounds of the for loop. ∑1 = N i=1 • Suppose we count the number of additions that are done. There is 1 addition per iteration of the loop, hence N additions in total. Nested Loops: Formally • Nested for loops translate into multiple summations, one for each for loop. N M N ∑∑2 = ∑2M = 2MN i=1 j=1 i=1 • Again, count the number of additions. The outer summation is for the outer for loop. Consecutive Statements: Formally • Add the running times of the separate blocks of your code

Design and Analysis of Algorithms module | PDF page 10 | lines 6-20

Analysis Rules: 1. We assume an arbitrary time unit. 2. Execution of one of the following operations takes time 1: • Assignment Operation • Single Input/output Operation • Single Boolean Operations • Single Arithmetic Operations • Function Return 3. Running time of a selection statement (if, switch) is the time for the condition evaluation plus the maximum of the running times for the individual clauses in the selection. 4. Loops: Running time for a loop is equal to the running time for the statements inside the loop multiplied by the number of iterations. The total running time of a statement inside a group of nested loops is the running time of the statements multiplied by the product of the sizes of all the loops. For nested loops, analyze inside out.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Question sources

Computer Science model exam collection · Question 23

Question 33 | SS120

Which claims about abstract algorithms are NOT true? Select all that apply. In C, platform considerations mean choosing realistic resource constraints, not making source language part of the definition.

A. Is step wise logical description of how to solve the problem

B. ls equivalent to the programming language code

C. Should be developed by considering the platforms on which it runs

D. Its abstract definition requires the details of a particular programming language.

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B, D: Is equivalent to the programming language code; Its abstract definition requires the details of a particular programming language.

An algorithm is a definite executable method independent of a particular source language. Equating it with programming-language code, or making that language's details essential to its abstract definition, is therefore false. A stepwise logical description is appropriate. Resource constraints and a platform-aware cost model can legitimately inform implementation choices without changing the underlying method's language independence. Thus B and D are the false claims.

Why each choice is right or wrong

A - Incorrect: A stepwise logical description is compatible with an algorithm.

B - Correct: The abstract method is not identical to a source-language implementation.

C - Incorrect: Realistic resource constraints may inform design, so C is not selected.

D - Correct: Programming-language details are not essential to the abstract method.

Primary note proof

Study-note section DAA01

> State inputs, output, constraints and empty-solution policy. Algorithms are finite, definite executable methods independent of source language; feasible satisfies constraints, optimal is best feasible. Conditional branches are legitimate control flow.

Study-note section DAA02

> Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 6-19

> • Finiteness: Algorithm must complete after a finite number of steps. • Definiteness: Each step must be clearly defined, having one and only one interpretation. At each point in computation, one should be able to tell exactly what happens next. • Sequence: Each step must have a unique defined preceding and succeeding step. The first step (start step) and last step (halt step) must be clearly noted. • Feasibility: It must be possible to perform each instruction. • Correctness: It must compute correct answer for all possible legal inputs. • Language Independence: It must not depend on any one programming language. • Completeness: It must solve the problem completely. • Effectiveness: It must be possible to perform each step exactly and in a finite

amount of time. • Efficiency: It must solve with the least amount of computational resources such as time and space. • Generality: Algorithm should be valid on all possible inputs • Input/Output: There must be a specified number of input values, and one or more result values.

Question sources

Computer Science model exam collection · Question 27

Question 34 | SS132

For integer n>=0, determine the tightest listed growth bound for this mathematical-integer pseudocode. Arithmetic, loop-control operations and each print operation have unit cost; ignore output-digit length.

```
function someFunction(n) {
for (i=0; i<10*n; i=i+1) {
print(n);
}
}
```

A. O(n)

B. $O(n^2)$

C. O(log n)

D. O(log(2n))

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: A. O(n)

The loop body runs 10n times. Each iteration performs a constant amount of loop-control work and one unit-cost print, while initialization and the final failed condition check contribute only constant additional work. The total therefore has linear growth. Quadratic is a looser upper bound, while logarithmic growth is too small. The explicit mathematical and output-cost model avoids syntax and representation-cost distractions.

Why each choice is right or wrong

A - Correct: A constant multiple of n iterations gives the tight linear bound.

B - Incorrect: Quadratic is a looser bound rather than the requested tightest one.

C - Incorrect: Logarithmic growth is too small for 10n body executions.

D - Incorrect: log(2n) differs from log n only by a constant and is also too small.

Primary note proof

Study-note section DAA06

Sequential costs add; independent nested loop counts multiply. A dependent loop requires summation. n iterations of a constant body give $\Theta(n)$; $\sum i=1..\ n\ i=n(n+1)/2$ gives $\Theta(n^2)$. Doubling i=1,2,4,... until i≥n takes ceil(log 2 n) iterations for n>1. A 10n loop is linear under constant-cost output.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 10 | lines 6-20

Analysis Rules: 1. We assume an arbitrary time unit. 2. Execution of one of the following operations takes time 1: • Assignment Operation • Single Input/output Operation • Single Boolean Operations • Single Arithmetic Operations • Function Return 3. Running time of a selection statement (if, switch) is the time for the condition evaluation plus the maximum of the running times for the individual clauses in the selection. 4. Loops: Running time for a loop is equal to the running time for the statements inside the loop multiplied by the number of iterations. The total running time of a statement inside a group of nested loops is the running time of the statements multiplied by the product of the sizes of all the loops. For nested loops, analyze inside out.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Question sources

Computer Science model exam collection · Question 39

Question 35 | SS141

Which of the following analysis mechanism defines the input for which the algorithm takes the least time (fastest time to complete)?

A. Standard Case

B. Average case

C. Worst case

D. Best case

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: Best case

The least runtime among inputs of a fixed size is best-case time, the fastest case in the p.14 three-case classification. It is not the average or worst case.

Why each choice is right or wrong

A - Incorrect: Standard case is not a defined category on p.14.

B - Incorrect: Average case concerns a distribution or average set, not a minimum (p.14).

C - Incorrect: Worst case takes the slowest input (p.14).

D - Correct: Best case is the minimum-time case intended by p.14.

Primary note proof

Study-note section DAA03

> At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

> In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Question sources

Computer Science model exam collection · Question 48

Question 36 | AM0344

Which described activity is outside analysis of an algorithm’s resource costs? Interpret dynamic analysis as measuring executed operations or storage.

A. Asymptotic analysis

B. Experimental analysis

C. Dynamic analysis

D. Graph connectivity analysis of the input data, without examining algorithm resource costs

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: Graph connectivity analysis of the input data, without examining algorithm resource costs

Asymptotic analysis studies resource growth, experimental analysis measures runs, and dynamic observation can measure executed operations or storage. The graph alternative asks only a mathematical connectivity question about input data, expressly without examining algorithm costs. It is outside the resource-analysis task as described, although graph-based representations can certainly support valid resource analysis in other contexts. This scope avoids an unsupported universal taxonomy.

Why each choice is right or wrong

A - Incorrect: Asymptotic analysis studies resource growth.

B - Incorrect: Experimental analysis measures resource costs of actual runs.

C - Incorrect: The stipulated dynamic analysis observes executed costs.

D - Correct: The stated connectivity-only task does not examine algorithm resource costs.

Primary note proof

Study-note section DAA02

> Experimental/dynamic measurement observes actual runs; asymptotic analysis abstracts growth; probabilistic analysis studies expectations. There is no universal taxonomy proving “graph analysis” or “dynamic analysis” is invalid. Auxiliary space excludes input; I/O complexity counts transfers. Representation affects costs; “data type complexity” is not a standard metric name.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

> 1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some

function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n * log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Design and Analysis of Algorithms module | PDF page 85 | lines 14-20

4.7. BASIC SEARCH AND TRAVERSAL TECHNIQUE DEFINING GRAPH: A graphs g consists of a set V of vertices (nodes) and a set E of edges (arcs). We write G=(V,E). V is a finite and non-empty set of vertices. E is a set of pair of vertices; these pairs are called as edges. Therefore, V(G) is read as V of G, is a set of vertices and E(G),read as E of G is a set of edges. An edge e= (v, w) is a pair of vertices v and w, and to be incident with v and w.

Question sources

Computer Science MCQ practice collection · Question 344

Question 37 | AM0345

What is the primary growth-related purpose of asymptotic analysis?

A. To measure the actual running time of an algorithm

B. To determine the best-case scenario for an algorithm

C. To analyze the behavior of an algorithm as the input size increases

D. To evaluate the correctness of an algorithm

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C: To analyze the behavior of an algorithm as the input size increases

Asymptotic analysis abstracts how resource use grows with input size. It can analyze best-, average- or worst-cost functions, but does not itself choose one input case or measure actual elapsed seconds. Those measurements depend on environment and workload. Correctness is a separate obligation to meet the specification. Hence C states the primary growth-related purpose among these choices.

Why each choice is right or wrong

A - Incorrect: Actual elapsed-time measurement is experimental.

B - Incorrect: Asymptotic analysis is not restricted to the best case.

C - Correct: It analyzes resource behavior as input size grows.

D - Incorrect: Growth bounds do not prove correctness.

Primary note proof

Study-note section DAA02

Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Study-note section DAA03

At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Variant 1 | Alternative version 1

Which are valid uses of asymptotic resource analysis? Select all that apply. In B, performance comparison means comparing growth under a common cost model, rather than predicting exact elapsed time.

A. To determine the actual running time of an algorithm

B. To compare the performance of different algorithms

C. To evaluate the correctness of an algorithm

D. To analyze the behavior of an algorithm as the input size increases

Answer B, D: To compare the performance of different algorithms; To analyze the behavior of an algorithm as the input size increases

Asymptotic analysis studies how cost grows with input size and can compare that scaling across algorithms under a common model. Thus both the growth description and the performance-comparison application are valid. It does not give actual elapsed seconds, and a favorable resource bound does not establish correctness. The explicit growth-based meaning of performance makes the two valid variant choices definite.

Why each choice is right or wrong

A - Incorrect: Actual elapsed time needs execution-environment measurements.

B - Correct: Comparing cost growth under a common model is a valid use.

C - Incorrect: Resource bounds do not establish correctness.

D - Correct: Behavior as input size grows is the direct purpose.

Primary note proof

Study-note section DAA02

> Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Study-note section DAA03

> At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Question sources

Computer Science MCQ practice collection · Question 345

Computer Science MCQ practice collection · Question 356

Question 38 | AM0346

Which notation expresses a general asymptotic upper bound on an algorithm’s worst-cost function, allowing equality up to constant factors and not requiring a matching lower bound?

A. Big O notation (O)

B. Theta notation (Θ)

C. Omega notation (Ω)

D. Small o notation (o)

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: Big O notation (O)

Big-O states an eventual constant-factor upper bound and allows the bounded function to have the same order as its comparator. Θ additionally requires a lower bound, Ω is a lower-bound relation, and little-o requires strictly slower growth. The worst-case function is selected separately by maximizing over equal-size inputs; Big-O itself does not mean worst case. The explicit upper-bound requirement uniquely identifies A.

Why each choice is right or wrong

A - Correct: O gives the requested non-strict upper bound.

B - Incorrect: Θ requires a matching lower bound as well.

C - Incorrect: Ω gives a lower bound.

D - Incorrect: Little-o requires strictly slower growth.

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, f∈O(g) means f(n)≤c·g(n) for all n≥n 0, with fixed c>0. Ω reverses that inequality; Θ requires both bounds. Little-o means f/g→0, and little-ω means f/g→∞ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Related reading (context, not primary note proof)

Study-note section DAA16

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

> In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

> I. Big-Oh Notation (O): It’s only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Data Structure and Algorithms module | PDF page 3 | lines 6-12

> II. Big-Omega Notation (Ω): provides an asymptotic lower bound; f(n)= Ω(g (n)) means that the growth rate of f(n) is greater than or equal to g(n). III. Theta Notation (Θ): If f(n)= Θ (g(n)), then g(n) is an asymptotically tight bound for f(n) i.e f(n) and g(n) have the same rate of growth. IV. Little-o Notation (o): f(n)=o(g(n)) means f(n) has less growth rate compared to g(n). V. Little-Omega Notation (ω): ω notation used to denote a lower bound that is not asymptotically tight.

Variant 1 | Alternative version 1

For ordinary array insertion sort on n distinct keys, count comparisons and shifts as constant-time operations. Which is the tightest listed Big-O upper bound for its worst case?

A. O(1)

B. O(n)

C. O(log n)

D. O(n^2)

Answer D: O(n^2)

In the worst input order, each inserted key shifts past the entire sorted prefix. The total work is proportional to $1+2+\cdots+(n-1)=\Theta(n^2)$. Thus $O(n^2)$ is the tightest listed worst-case upper bound. Constant, logarithmic and linear bounds are too small. The supplied algorithm and cost model make this growth-class variant answerable without inventing a universal worst-case rate.

Why each choice is right or wrong

A - Incorrect: Constant cannot bound the quadratic worst-case work.

B - Incorrect: Linear cannot bound the summed prefix shifts.

C - Incorrect: Logarithmic is too small for quadratic work.

D - Correct: Insertion sort has $\Theta(n^2)$ worst work, so $O(n^2)$ is tight.

Primary note proof

Study-note section DAA16

> Insertion shifts larger sorted-prefix keys: stable with strict comparisons, Θ(n) best and $\Theta(n^2)$ worst.

Related reading (context, not primary note proof)

Study-note section DAA04

Question sources

Computer Science MCQ practice collection · Question 346

Computer Science MCQ practice collection · Question 357

Computer Science MCQ practice collection · Question 386

Question 39 | AM0347

For binary search on a nonempty sorted random-access array, what is the tight best-case time when the key may be at the initial midpoint? Use constant-time comparisons.

A. $\Theta(1)$

B. $\Theta(\log n)$

C. $\Theta(n)$

D. $\Theta(n^2)$

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: Θ(1)

If the sought key is at the first midpoint, binary search finds it after one constant-time comparison and returns. This is $\Theta(1)$ best-case time, regardless of the array length. The logarithmic bound describes its worst case, not this fastest legal input at fixed size. The linear and quadratic alternatives are likewise not tight for the midpoint-match case.

Why each choice is right or wrong

A - Correct: An initial midpoint match takes constant work.

B - Incorrect: Logarithmic describes worst-case search depth, not the tight best case.

C - Incorrect: Linear is not tight for a one-comparison match.

D - Incorrect: Quadratic is not tight for a one-comparison match.

Primary note proof

Study-note section DAA13

> Binary search requires sorted random-access data. Maintain inclusive low/high bounds, compute mid=low+(high−low)//2, and replace high with mid−1 or low with mid+1 according to the comparison. Stop when found or low>high. $T(n)=T(n/2)+\Theta(1)$: worst $\Theta(\log n)$, best $\Theta(1)$; iterative auxiliary space $\Theta(1)$, recursive $\Theta(\log n)$.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

> In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

> I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Question sources

Computer Science MCQ practice collection · Question 347

Question 40 | AM0348

For ordinary array insertion sort on n distinct keys, with constant-time comparisons and shifts, what is the tight worst-case running time?

A. $\Theta(\log n)$

B. $\Theta(n)$

C. $\Theta(n \log n)$

D. $\Theta(n^2)$

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: $\Theta(n^2)$

A descending input makes each inserted key move across every earlier key in the sorted prefix. The work sums as $1+2+\cdots+(n-1)$, which is $\Theta(n^2)$. This establishes the tight worst case for the specified insertion-sort implementation. The smaller alternatives cannot upper-bound that quadratic work, and the answer is not a general claim about worst-case complexity of every algorithm.

Why each choice is right or wrong

A - Incorrect: Logarithmic is too small for the summed shifts.

B - Incorrect: Linear is too small for the summed shifts.

C - Incorrect: n log n is too small for quadratic worst-case shifts.

D - Correct: The sum of prefix lengths gives quadratic work.

Primary note proof

Study-note section DAA16

Insertion shifts larger sorted-prefix keys: stable with strict comparisons, $\Theta(n)$ best and $\Theta(n^2)$ worst. Optimized bubble sort swaps adjacent inversions and stops after a no-swap pass: stable, $\Theta(n)$ best, $\Theta(n^2)$ worst. Selection repeatedly finds a suffix minimum: $\Theta(n^2)$ comparisons in every order, normally unstable. These use $\Theta(1)$ auxiliary array space.

Study-note section DAA06

Sequential costs add; independent nested loop counts multiply. A dependent loop requires summation. n iterations of a constant body give $\Theta(n)$; $\sum i=1..\ n\ i=n(n+1)/2$ gives $\Theta(n^2)$. Doubling i=1,2,4,... until i≥n takes ceil(log 2 n) iterations for n>1. A 10n loop is linear under constant-cost output.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

> I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Question sources

Computer Science MCQ practice collection · Question 348

Question 41 | AM0349

Which of the following time complexities represents an algorithm with logarithmic growth?

A. O(1)

B. O(log n)

C. O(n)

D. O(n^2)

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer B: O(log n)

Binary search takes time proportional to log_2 n. B matches logarithmic growth.

Why each choice is right or wrong

A - Incorrect: 1 is a constant, not a logarithm (p.14 growth comparison).

B - Correct: log n is the logarithmic class named on p.14 and instantiated in cross-module p.3.

C - Incorrect: n is linear rather than logarithmic (p.14).

D - Incorrect: n^2 is quadratic rather than logarithmic (p.14).

Primary note proof

Study-note section DAA05

> Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the

size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching • Works on ordered/sorted list • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list • The computational time for this algorithm is proportional to log2 n.

Question sources

Computer Science MCQ practice collection · Question 349

Computer Science MCQ practice collection · Question 361

Question 42 | AM0350

Which of the following time complexities represents an algorithm with linear growth?

A. O(1)

B. O(log n)

C. O(n)

D. O(n^2)

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C: O(n)

Linear growth means work proportional to input size n; sequential search takes O(n) time. The growth classes n, log n and n^2 are distinct.

Why each choice is right or wrong

A - Incorrect: 1 stays bounded instead of increasing proportionally to n (p.14).

B - Incorrect: log n increases more slowly than n (p.14 growth classes).

C - Correct: O(n) is the linear class illustrated by sequential search (cross-module p.3).

D - Incorrect: n^2 scales with the square rather than the first power (p.14).

Primary note proof

Study-note section DAA05

> Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Study-note section DAA06

> n iterations of a constant body give Θ(n);

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

> Linear/Sequential searching • Works with unsorted List • Loop through the list starting at the first element • Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Computer Science MCQ practice collection · Question 350

Question 43 | AM0351

Which tight running-time growth rates are strictly superlinear, meaning f(n)/n→∞? Select all that apply.

A. Θ(log n)

B. Θ(n)

C. Θ(n log n)

D. Θ(n^2)

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C, D: Θ(n log n); Θ(n^2)

For n log n, dividing by n gives log n, which grows without bound. For n^2, the ratio is n, also unbounded. Both are therefore superlinear. The logarithmic representative is sublinear and n itself has constant ratio one, so neither meets the strict condition. Using tight rates prevents a loose upper-bound label from being mistaken for an algorithm's actual growth.

Why each choice is right or wrong

A - Incorrect: log n grows more slowly than n.

B - Incorrect: n is linear, with ratio one.

C - Correct: The ratio to n is log n and diverges.

D - Correct: The ratio to n is n and diverges.

Primary note proof

Study-note section DAA05

> Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

> I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Question sources

Computer Science MCQ practice collection · Question 351

Question 44 | AM0352

What is the purpose of experimental analysis in algorithm analysis?

A. To measure the actual running time of an algorithm

B. To determine the best-case scenario for an algorithm

C. To analyze the behavior of an algorithm as the input size increases

D. To evaluate the correctness of an algorithm

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: To measure the actual running time of an algorithm

The module distinguishes analytical performance analysis from conducting experiments for performance measurement (PDF p.4). Experimental analysis therefore measures actual performance on chosen executions rather than deriving a limiting rate or proving correctness.

Why each choice is right or wrong

A - Correct: Direct runtime measurement is the performance-measurement purpose (p.4).

B - Incorrect: An experiment might include favorable inputs but does not inherently determine a universal best case (p.4).

C - Incorrect: Growth analysis is analytical; experiments can sample sizes but are not the defining limiting argument (p.4).

D - Incorrect: Correctness tests can accompany experiments, but the stated performance-measurement purpose is separate (p.4).

Primary note proof

Study-note section DAA02

> Experimental/dynamic measurement observes actual runs; asymptotic analysis abstracts growth; probabilistic analysis studies expectations. There is no universal taxonomy proving “graph analysis” or “dynamic analysis” is invalid. Auxiliary space excludes input; I/O complexity counts transfers. Representation affects costs; “data type complexity” is not a standard metric name.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Variant 1 | Alternative version 1

What is the purpose of experimental analysis in algorithm analysis?

A. To determine the best-case time complexity of an algorithm.

B. To compare algorithms based on their theoretical analysis.

C. To evaluate the correctness of an algorithm.

D. To measure the actual performance of an algorithm on real data.

Answer D: To measure the actual performance of an algorithm on real data.

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Incorrect: Experiments do not by themselves prove the minimum cost for every input size (p.4 distinction).

B - Incorrect: Comparison based on theoretical analysis uses the analytical approach (p.4).

C - Incorrect: Functional correctness is not the stated performance-measurement objective (p.4).

D - Correct: Measured performance on real data is the experimental approach (p.4).

Primary note proof

Study-note section DAA02

Experimental/dynamic measurement observes actual runs; asymptotic analysis abstracts growth; probabilistic analysis studies expectations. There is no universal taxonomy proving “graph analysis” or “dynamic analysis” is invalid. Auxiliary space excludes input; I/O complexity counts transfers. Representation affects costs; “data type complexity” is not a standard metric name.

Question sources

Computer Science MCQ practice collection · Question 352

Computer Science MCQ practice collection · Question 362

Question 45 | AM0353

Which offered phrase is not a standard resource-metric name in the course's time/space/I-O cost model? This does not ask whether representation affects cost.

A. Space complexity

B. Auxiliary space

C. Input-output complexity

D. Data type complexity

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer D: Data type complexity

Space complexity measures storage, auxiliary space isolates additional storage beyond input, and I/O complexity counts transfers. These are resource metrics in the stated model. Data type complexity is not the standard metric name used for representation effects; those effects are accounted for through arithmetic, storage or bit-operation costs. Therefore D is the terminology exception without asserting that data representation is irrelevant.

Why each choice is right or wrong

A - Incorrect: Space complexity is a standard storage metric.

B - Incorrect: Auxiliary space is a standard additional-storage metric.

C - Incorrect: I/O complexity is a standard transfer metric.

D - Correct: Data type complexity is not the standard metric name in this model.

Primary note proof

Study-note section DAA02

> Experimental/dynamic measurement observes actual runs; asymptotic analysis abstracts growth; probabilistic analysis studies expectations. There is no universal taxonomy proving "graph analysis" or "dynamic analysis" is invalid. Auxiliary space excludes input; I/O complexity counts transfers. Representation affects costs; "data type complexity" is not a standard metric name.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

> The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to

be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Question sources

Computer Science MCQ practice collection · Question 353

Question 46 | AM0354

What is the purpose of algorithm analysis in computer science?

A. To design algorithms for specific problems

B. To implement algorithms using programming languages

C. To analyze the efficiency and performance of algorithms

D. To test algorithms for correctness

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C: To analyze the efficiency and performance of algorithms

Performance analysis estimates computer memory and time requirements (PDF p.4), and studying those requirements helps choose between algorithms (p.6). Thus efficiency/performance is the direct purpose asked for.

Why each choice is right or wrong

A - Incorrect: Design creates a method; p.4 analysis estimates resources of a method.

B - Incorrect: Implementation expresses a method in code; p.4 analysis estimates performance.

C - Correct: This directly matches the memory-and-time performance definition (p.4).

D - Incorrect: Correctness is necessary but is distinct from the p.4 performance analysis objective.

Primary note proof

Study-note section DAA02

Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to

bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Question sources

Computer Science MCQ practice collection · Question 354

Question 47 | AM0355

Which can be legitimate techniques for analyzing algorithm resource costs? Select all that apply. Dynamic means observing actual executions; empirical means measuring runs.

A. Asymptotic analysis

B. Empirical analysis

C. Dynamic analysis

D. Probabilistic analysis

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A, B, C, D: Asymptotic analysis; Empirical analysis; Dynamic analysis; Probabilistic analysis

Asymptotic reasoning studies growth, empirical measurements observe actual runs, and dynamic analysis under the stated meaning also observes execution costs. Probabilistic analysis studies expected cost under a distribution or internal randomness. All four can therefore be legitimate resource-analysis techniques. Empirical and dynamic may overlap; they do not have to be disjoint categories for their methods to be useful.

Why each choice is right or wrong

A - Correct: Asymptotic analysis studies cost growth.

B - Correct: Empirical analysis measures actual runs.

C - Correct: Dynamic observation measures actual execution costs under the given definition.

D - Correct: Probabilistic analysis studies expectations.

Primary note proof

Study-note section DAA02

> Experimental/dynamic measurement observes actual runs; asymptotic analysis abstracts growth; probabilistic analysis studies expectations. There is no universal taxonomy proving “graph analysis” or “dynamic analysis” is invalid. Auxiliary space excludes input; I/O complexity counts transfers. Representation affects costs; “data type complexity” is not a standard metric name.

Study-note section DAA03

> At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

> 1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

> In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the

worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Question sources

Computer Science MCQ practice collection · Question 355

Question 48 | AM0358

What does a time complexity of O(1) indicate?

A. The algorithm runs in constant time.

B. The algorithm runs in logarithmic time.

C. The algorithm runs in linear time.

D. The algorithm runs in quadratic time.

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: The algorithm runs in constant time.

A constant upper bound O(1) does not grow with input size; it is the constant K at the start of the visually inspected growth ordering in Data Structures and Algorithms PDF p.3. The assigned PDF p.14 motivates classifying by growth rather than exact operation count.

Why each choice is right or wrong

A - Correct: Constant time has a fixed upper bound independent of n (cross-module p.3).

B - Incorrect: Logarithmic time grows with n, unlike 1 (assigned p.14).

C - Incorrect: Linear time grows proportionally to n (assigned p.14).

D - Incorrect: Quadratic time grows as n^2 (assigned p.14).

Primary note proof

Study-note section DAA04

For eventually nonnegative f, g, f∈O(g) means f(n)≤c·g(n) for all n≥n 0, with fixed c>0. Ω reverses that inequality; Θ requires both bounds. Little-o means f/g→0, and little-ω means f/g→∞ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Study-note section DAA05

Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Question sources

Computer Science MCQ practice collection · Question 358

Question 49 | AM0359

Which of the following time complexities indicates the most efficient algorithm?

A. O(1)

B. O(log n)

C. O(n)

D. O(n^2)

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer A: O(1)

When comparing the listed representative growth rates for the same task and cost model as n becomes large, constant growth is smallest. The visually inspected related-module PDF p.3 ordering places K before log n, n and n^2; assigned p.14 focuses on large-input growth. This is not a guarantee about clock time for every finite input.

Why each choice is right or wrong

A - Correct: Constant is the smallest representative listed growth (cross-module p.3 diagram).

B - Incorrect: log n grows beyond any fixed constant (p.14 growth interpretation).

C - Incorrect: n grows faster than constant and logarithmic rates (p.14).

D - Incorrect: n^2 grows fastest among these listed rates (p.14).

Primary note proof

Study-note section DAA05

> Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

> I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What

happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Question sources

Computer Science MCQ practice collection · Question 359

Question 50 | AM0360

What does a time complexity of O(n^2) signify?

A. The algorithm runs in linear time.

B. The algorithm runs in logarithmic time.

C. The algorithm runs in quadratic time.

D. The algorithm runs in exponential time.

Hint

Identify the specification and cost model, then apply the relevant bound, invariant or strategy rule.

Answer C: The algorithm runs in quadratic time.

The exponent 2 identifies the quadratic representative growth class n^2, listed on PDF p.14. Strictly O(n^2) is an upper bound (cross-module p.3), so it does not prove that an algorithm actually requires quadratic time. C is the conventional class-name answer.

Why each choice is right or wrong

A - Incorrect: Linear is the n class, not the name of n^2 (p.14).

B - Incorrect: Logarithmic is the log n class, distinct from n^2 (p.14).

C - Correct: Quadratic names the squared-input growth class (p.14).

D - Incorrect: Exponential uses a growing exponent, such as 2^n, separately listed on p.14.

Primary note proof

Study-note section DAA04

For eventually nonnegative f, g, f∈O(g) means f(n)≤c·g(n) for all n≥n 0, with fixed c>0. Ω reverses that inequality; Θ requires both bounds. Little-o means f/g→0, and little-ω means f/g→∞ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Study-note section DAA05

Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Question sources

Computer Science MCQ practice collection · Question 360

Question 51 | AM0363

In fixed unit-cost RAM analysis of operation count and storage as functions of input size, which factor is abstracted away rather than being a parameter of the analysis?

A. Time complexity

B. Space complexity

C. Input size

D. Hardware configuration

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer D: Hardware configuration

The fixed RAM model counts operations and storage relative to input size. Time complexity, space complexity and input size are therefore central to the analysis. The particular hardware configuration is abstracted away by the stipulated unit costs. It would matter for actual elapsed measurements, alongside compiler, cache, I/O and load, but that is a different analysis setting from the one explicitly requested.

Why each choice is right or wrong

A - Incorrect: Operation-count time complexity is a central resource.

B - Incorrect: Space complexity is a central resource.

C - Incorrect: Costs are expressed as functions of input size.

D - Correct: The fixed unit-cost abstraction hides the specific hardware configuration.

Primary note proof

Study-note section DAA02

> Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

> 1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

> The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Design and Analysis of Algorithms module | PDF page 10 | lines 6-20

> Analysis Rules: 1. We assume an arbitrary time unit. 2. Execution of one of the following operations takes time 1: • Assignment Operation • Single Input/output Operation • Single Boolean

Operations • Single Arithmetic Operations • Function Return 3. Running time of a selection statement (if, switch) is the time for the condition evaluation plus the maximum of the running times for the individual clauses in the selection. 4. Loops: Running time for a loop is equal to the running time for the statements inside the loop multiplied by the number of iterations. The total running time of a statement inside a group of nested loops is the running time of the statements multiplied by the product of the sizes of all the loops. For nested loops, analyze inside out.

Design and Analysis of Algorithms module | PDF page 6 | lines 18-30

Components of space complexity: - the space needed by a program has the following components: • Instruction space: - is the amount of space needed to store the compiled version of the program instructions. The amount of instruction space that is needed depends on factors such as: - The compiler used to compile the program into machine code. - The compiler options in effect at time of compilation. - The target computer. • Data space: - is the space needed to store all constant and variable values. Data space has two components: i. Space needed by constants (e.g. the numbers 0, 1, 2...) and simple variables. ii. Space needed by dynamically allocated objects such as arrays and class instances. • Environmental stack space:- is the space used to save information needed to resume execution of partially completed methods (functions). For example, if method func1 invokes or calls method func2, then we must at least save a pointer to the instruction of func1 to be executed when func2

Question sources

Computer Science MCQ practice collection · Question 363

Question 52 | AM0364

Which technique solves smaller largely independent instances recursively and then combines their results, without relying on reuse of overlapping states?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer A: Divide and conquer

Divide and conquer separates a problem into smaller largely independent instances, solves them and combines the results. Dynamic programming instead emphasizes defined reusable states when subproblems overlap. Greedy commits to safe local choices, while backtracking explores and retracts partial candidates. The independence and combination conditions distinguish the requested technique from these other ways of reducing or extending a problem.

Why each choice is right or wrong

A - Correct: Independent recursive subinstances and combination define divide and conquer.

B - Incorrect: Greedy is defined by safe irrevocable local choices.

C - Incorrect: DP emphasizes reusable subproblem states rather than this independent structure.

D - Incorrect: Backtracking is defined by candidate extension and retraction.

Primary note proof

Study-note section DAA12

Divide and conquer solves smaller largely independent instances and combines results. DP reuses a defined set of subproblem states. Greedy commits to a feasible local choice only when a safety argument supports it. Backtracking grows/retracts candidates when constraints fail; branch and bound also rejects candidates whose best possible objective cannot beat an incumbent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 17 | lines 5-9

2.1 Divide and Conquer Algorithm In this approach, we solve a problem recursively by applying 3 steps 1. DIVIDE-break the problem into several sub problems of smaller size. 2. CONQUER-solve the problem recursively. 3. COMBINE-combine these solutions to create a solution to the original problem.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Design and Analysis of Algorithms module | PDF page 95 | lines 3-17

5. Introduction to Backtracking 5.1 Backtracking: • It is one of the most general algorithm design techniques. • Many problems which deal with searching for a set of solutions or for a optimal solution satisfying some constraints can be solved using the backtracking formulation. • To apply backtracking method, the desired solution must be expressible as an n- tuple (x1…xn) where xi is chosen from some finite set Si. • The problem is to find a vector, which maximizes or minimizes a criterion function P(x1….xn). • The major advantage of this method is, once we know that a partial vector (x1,…xi) will not lead to an optimal solution that (mi+1…….mn) possible test

vectors may be ignored entirely. • Many problems solved using backtracking require that all the solutions satisfy a complex set of constraints.

Question sources

Computer Science MCQ practice collection · Question 364

Question 53 | AM0365

Which algorithm design technique guarantees finding the optimal solution at each step but may not lead to the overall optimal solution?

A. Greedy algorithm

B. Divide and conquer

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer A: Greedy algorithm

Greedy makes the best available local choice based on current information (PDF p.27), and the module explicitly warns that such choices can fail to yield a global optimum (p.30). “Optimal at each step” must mean locally best, not globally guaranteed.

Why each choice is right or wrong

A - Correct: Local short-sighted choices and possible global failure match pp.27,30.

B - Incorrect: Independent splitting and combining does not define a locally best choice (p.27 contrast).

C - Incorrect: Dynamic programming retains subproblem alternatives rather than irrevocably picking only the local best (p.30 contrast).

D - Incorrect: Backtracking revisits choices, unlike non-recoverable greedy choices on p.30.

Primary note proof

Study-note section DAA19

A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it. Optimal substructure alone does not prove this safe. An exchange proof inserts the greedy choice into an optimum without worsening cost; recurse. MST cut proofs use this exchange.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 27 | lines 7-18

• Greedy method is the most straightforward designed technique. • As the name suggest they are short sighted in their approach taking decision on the basis of the information immediately at the hand without worrying about the effect these decisions may have in the future. DEFINITION: • A problem with N inputs will have some constraints. Any subsets that satisfy these constraints are called a feasible solution. • A feasible solution that either maximize can minimize a given objectives function is called an optimal solution. Greedy is a strategy that works well on optimization problems with the following characteristics: • Greedy-choice property: A global optimum can be arrived at by selecting a local

Design and Analysis of Algorithms module | PDF page 30 | lines 10-14

Greedy algorithms sometimes fail to produce the optimal solution, and may even produce the unique worst possible solution. One example is the travelling salesman problem. Greedy algorithms can be characterized as being 'short sighted', and as 'non-recoverable'. They are ideal only for problems which have 'optimal substructure'. Despite this, greedy algorithms are best suited for simple problems (e.g. giving change). It is important, however, to note that

Question sources

Computer Science MCQ practice collection · Question 365

Question 54 | AM0366

Which algorithm design technique is used to solve problems by exploring all possible solutions?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer D: Backtracking

Backtracking generates solution tuples and explores choices with constraints, returning to earlier positions when needed (PDF pp.95-96). It can enumerate all valid solutions while pruning branches proved unpromising, so “all possible solutions” should not be read as requiring every invalid tuple to be visited.

Why each choice is right or wrong

A - Incorrect: Divide and conquer combines independent solved parts rather than enumerating candidate tuples (pp.95-96 contrast).

B - Incorrect: Greedy follows one selection sequence rather than revisiting alternatives (pp.95-96 contrast).

C - Incorrect: Dynamic programming caches subproblems rather than being the tuple-enumeration method shown on p.96.

D - Correct: The p.96 schema generates all solutions and backtracks over choices.

Primary note proof

Study-note section DAA36

Represent a partial assignment x[1.. k]. For each allowed next choice: assign it, test whether an extension remains possible, recurse if promising, and undo the assignment on return. Report complete feasible candidates; stop after one only if the task requests existence/a single answer. Explicit constraints restrict each variable’s domain; implicit constraints relate choices.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 95 | lines 3-17

5. Introduction to Backtracking 5.1 Backtracking: • It is one of the most general algorithm design techniques. • Many problems which deal with searching for a set of solutions or for a optimal solution satisfying some constraints can be solved using the backtracking formulation. • To apply backtracking method, the desired solution must be expressible as an n- tuple (x1…xn) where xi is chosen from some finite set Si. • The problem is to find a vector, which maximizes or minimizes a criterion function P(x1….xn). • The major advantage of this method is, once we know that a partial vector (x1,…xi) will not lead to an optimal solution that (mi+1…….mn) possible test vectors may be ignored entirely. • Many problems solved using backtracking require that all the solutions satisfy a complex set of constraints.

Design and Analysis of Algorithms module | PDF page 96 | lines 3-21

Algorithm Backtracking (n) // This schema describes the backtracking procedure. All solutions are generated in X[1:n] //and printed as soon as they are determined. { k=1; While (k ≠ 0) do { if (there remains all untried X[k] ∈ T (X[1],[2],…..X[k-1]) and Bk (X[1],…..X[k])) is true) then { if(X[1],……X[k])is the path to the answer node) Then write(X[1:k]); k=k+1; //consider the next step. } else k=k-1; //consider backtracking to the previous set. }} • All solutions are generated in X[1:n] and printed as soon as they are determined. • T(X[1]…..X[k-1]) is all possible values of X[k] gives that X[1],……..X[k-1] have already been chosen.

Question sources

Computer Science MCQ practice collection · Question 366

Question 55 | AM0367

Which algorithm design technique is suitable for problems with overlapping subproblems and optimal substructure?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: Dynamic programming

Shared sub-subproblems motivate saving results once (PDF p.48); optimal substructure makes an optimal solution a combination of optimal subsolutions (pp.28,49). Together these are the properties for the module's dynamic-programming optimization method.

Why each choice is right or wrong

A - Incorrect: Independent subproblems distinguish divide and conquer in p.48.

B - Incorrect: Optimal substructure alone does not prove the greedy-choice property (pp.28,49).

C - Correct: Caching shared subproblem solutions with optimal substructure is exactly the method on pp.48-49.

D - Incorrect: Backtracking searches constrained tuples; it need not exploit repeated subproblems via a table (p.48 contrast).

Primary note proof

Study-note section DAA25

> Define state meaning, choices, recurrence, bases and dependency order. Prove replacing a subsolution by a better one improves the full solution; otherwise state information may be missing. Show the recurrence covers all legal final choices and generates feasible candidates, then induct in dependency order. Store choices for reconstruction.

Study-note section DAA26

> Memoization recursively checks a cache before solving a state and stores its result. Tabulation orders dependencies explicitly; chosen domains may exclude irrelevant states. Cache keys include all relevant parameters; recursion/locality/sparsity affect implementation cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

> Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Design and Analysis of Algorithms module | PDF page 28 | lines 3-4

> • Optimal substructure: An optimal solution to the problem contains an optimal solution to sub problems.

Design and Analysis of Algorithms module | PDF page 49 | lines 2-12

> The Principle of Optimality The dynamic programming relies on a principle of optimality. This principle states that in an optimal sequence of decisions or choices, each subsequence must also be optimal. For example, in matrix chain multiplication problem, not only the value we are interested in is optimal but all the other entries in the table are also represent optimal. The principle can be related as follows: the optimal solution to a problem is a combination of optimal solutions to some of its sub problems. The difficulty in turning the principle of optimally into an algorithm is that it is not usually obvious which sub problems are relevant to the problem under consideration. General method-multistage graphs-all pair shortest path algorithm-Optimal Binary Search trees- 0/1 knapsack Problem-Reliability design-traveling salesman problem

Question sources

Computer Science MCQ practice collection · Question 367

Computer Science MCQ practice collection · Question 378

Question 56 | AM0368

Which technique builds an optimum by repeatedly making a locally attractive feasible choice, never undoing it, when an exchange or safety proof justifies each commitment?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer B: Greedy algorithm

Greedy algorithms repeatedly commit to locally attractive feasible choices without later undoing them. An exchange or related safety proof is needed to show that these commitments can form an optimum. Dynamic programming compares reusable states, backtracking revises candidates, and divide and conquer combines independent subinstances. The explicit safety and irrevocability conditions distinguish greedy from the broad idea of an optimal choice sequence.

Why each choice is right or wrong

A - Incorrect: Divide and conquer relies on independent decomposition and combination.

B - Correct: Safe irrevocable local commitments define greedy.

C - Incorrect: DP evaluates state alternatives rather than requiring irrevocable local choices.

D - Incorrect: Backtracking undoes choices, contrary to the stated rule.

Primary note proof

Study-note section DAA19

> A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it. Optimal substructure alone does not prove this safe. An exchange proof inserts the greedy choice into an optimum without worsening cost; recurse. MST cut proofs use this exchange.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 27 | lines 7-18

> • Greedy method is the most straightforward designed technique. • As the name suggest they are short sighted in their approach taking decision on the basis of the information immediately at the hand without worrying about the effect these decisions may have in the future. DEFINITION: • A problem with N inputs will have some constraints. Any subsets that satisfy these constraints are called a feasible solution. • A feasible solution that either maximize can minimize a given objectives function is called an optimal solution. Greedy is a strategy that works well on optimization problems with the following characteristics: • Greedy-choice property: A global optimum can be arrived at by selecting a local

Design and Analysis of Algorithms module | PDF page 49 | lines 2-12

> The Principle of Optimality The dynamic programming relies on a principle of optimality. This principle states that in an optimal sequence of decisions or choices, each subsequence must also be optimal. For example, in matrix chain multiplication problem, not only the value we are interested in is optimal but all the other entries in the table are also represent optimal. The principle can be related as follows: the optimal solution to a problem is a combination of optimal solutions to some of its sub problems. The difficulty in turning the principle of optimally into an algorithm is that it is not usually obvious which sub problems are relevant to the problem under consideration. General method-multistage graphs-all pair shortest path algorithm-Optimal Binary Search trees- 0/1 knapsack Problem-Reliability design-traveling salesman problem

Design and Analysis of Algorithms module | PDF page 49 | lines 27-31

• The essential difference between the greedy method and dynamic programming is that the greedy method only one decision sequence is ever generated. • In dynamic programming, many decision sequences may be generated. However, sequences containing sub-optimal sub-sequences cannot be optimal and so will not be generated.

Question sources

Computer Science MCQ practice collection · Question 368

Question 57 | AM0369

Which algorithm design technique is suitable for solving problems with optimal substructure but does not involve overlapping subproblems?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer A: Divide and conquer is the supported independent-subproblem strategy.

DAA12 characterizes divide and conquer by smaller largely independent instances. The stated absence of overlapping subproblems fits that structure. Optimal substructure alone does not establish greedy safety, as DAA19 warns, while DP's main reuse motivation is overlap. Backtracking needs a candidate-and-constraint search description absent here. A is therefore the supported textbook structural match, rather than a proof that only it can ever apply.

Why each choice is right or wrong

A - Correct: Independent smaller instances with combinable optimal results fit divide and conquer.

B - Incorrect: Optimal substructure alone does not prove a safe greedy local choice.

C - Incorrect: Without overlapping states, the stated premise does not motivate DP reuse.

D - Incorrect: No candidate retraction or constraint-based enumeration is specified to identify backtracking.

Primary note proof

Study-note section DAA12

Divide and conquer solves smaller largely independent instances and combines results. DP reuses a defined set of subproblem states. Greedy commits to a feasible local choice only when a safety argument supports it. Backtracking grows/retracts candidates when constraints fail;

branch and bound also rejects candidates whose best possible objective cannot beat an incumbent.

Study-note section DAA19

Optimal substructure alone does not prove this safe.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Design and Analysis of Algorithms module | PDF page 27 | lines 7-18

• Greedy method is the most straightforward designed technique. • As the name suggest they are short sighted in their approach taking decision on the basis of the information immediately at the hand without worrying about the effect these decisions may have in the future. DEFINITION: • A problem with N inputs will have some constraints. Any subsets that satisfy these constraints are called a feasible solution. • A feasible solution that either maximize can minimize a given objectives function is called an optimal solution. Greedy is a strategy that works well on optimization problems with the following characteristics: • Greedy-choice property: A global optimum can be arrived at by selecting a local

Design and Analysis of Algorithms module | PDF page 28 | lines 3-4

• Optimal substructure: An optimal solution to the problem contains an optimal solution to sub problems.

Question sources

Computer Science MCQ practice collection · Question 369

Question 58 | AM0370

Which technique explores a series of candidate choices, retracting a choice when constraints show that its partial candidate cannot extend to a solution?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer D: Backtracking

Backtracking extends a partial assignment, tests whether a feasible completion remains possible, and undoes the assignment when returning to try alternatives. That matches the stated constraint-driven retraction. Greedy keeps its choices, DP stores reusable state values, and divide and conquer combines independent instances. Merely having a series of choices would not distinguish them; the explicit undo-and-retry mechanism does.

Why each choice is right or wrong

A - Incorrect: Independent decomposition is not the described candidate retraction.

B - Incorrect: Greedy does not undo committed choices.

C - Incorrect: DP is characterized by reuse of states rather than this search rule.

D - Correct: Backtracking extends, checks, retracts and tries alternatives.

Primary note proof

Study-note section DAA36

> Represent a partial assignment x[1.. k]. For each allowed next choice: assign it, test whether an extension remains possible, recurse if promising, and undo the assignment on return. Report complete feasible candidates; stop after one only if the task requests existence/a single answer. Explicit constraints restrict each variable’s domain; implicit constraints relate choices.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 48 | lines 17-22

> Dynamic programming is a stage-wise search method suitable for optimization problems whose solutions may be viewed as the result of a sequence of decisions. The most attractive property of this strategy is that during the search for a solution it avoids full enumeration by pruning early partial decision solutions that cannot possibly lead to optimal solution. In many practical situations, this strategy hits the optimal solution in a polynomial number of decision steps. However, in the worst case, such a strategy may end up performing full enumeration.

Design and Analysis of Algorithms module | PDF page 95 | lines 3-17

> 5. Introduction to Backtracking 5.1 Backtracking: • It is one of the most general algorithm design techniques. • Many problems which deal with searching for a set of solutions or for a optimal solution satisfying some constraints can be solved using the backtracking formulation. • To apply backtracking method, the desired solution must be expressible as an n- tuple (x1…xn) where xi is chosen from some finite set Si. • The problem is to find a vector, which maximizes or minimizes a criterion function P(x1….xn). • The major advantage of this method is, once we know that a partial vector (x1,…xi) will not lead to an optimal solution that (mi+1…….mn) possible test vectors may be ignored entirely. • Many problems solved using backtracking require that all the solutions satisfy a complex set of constraints.

Design and Analysis of Algorithms module | PDF page 49 | lines 27-31

> • The essential difference between the greedy method and dynamic programming is that the greedy method only one decision sequence is ever generated. • In dynamic programming, many decision sequences may be generated. However, sequences containing sub-optimal sub-sequences cannot be optimal and so will not be generated.

Question sources

Computer Science MCQ practice collection · Question 370

Question 59 | AM0371

Which listed techniques can support exact optimization when their required structural or exhaustive-search conditions hold? Select all that apply; consider maximum subarray, fractional knapsack, 0/1 knapsack and complete tour enumeration as examples.

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer A, B, C, D: Divide and conquer; Greedy algorithm; Dynamic programming; Backtracking

Divide and conquer optimizes maximum subarray by comparing left, right and crossing results. Greedy optimizes fractional knapsack under divisibility, DP optimizes 0/1 knapsack states, and complete backtracking can enumerate tours and retain the cheapest. All four techniques therefore have exact optimization applications under their own conditions. None becomes universally appropriate merely because the task asks for an optimum.

Why each choice is right or wrong

A - Correct: Maximum-subarray divide and conquer is an exact optimizer.

B - Correct: Density-greedy is exact for divisible fractional knapsack.

C - Correct: 0/1 knapsack DP computes an exact optimum.

D - Correct: Complete tour enumeration can retain the exact minimum.

Primary note proof

Study-note section DAA18

> Divide-and-conquer compares the best left, best right and best crossing result.

Study-note section DAA20

> For positive weights/profits and divisible items, sort by decreasing value/weight, take complete items while they fit, then the fitting fraction of the next.

Study-note section DAA29

> For integer capacity W, K[i, w] is maximum value using the first i items within capacity w.

Study-note section DAA33

> Greedy nearest-neighbor is a heuristic without a general exactness guarantee; backtracking and branch and bound can also solve exact TSP.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 27 | lines 7-18

> • Greedy method is the most straightforward designed technique. • As the name suggest they are short sighted in their approach taking decision on the basis of the information immediately at the hand without worrying about the effect these decisions may have in the future. DEFINITION: • A problem with N inputs will have some constraints. Any subsets that satisfy these constraints are called a feasible solution. • A feasible solution that either maximize can minimize a given objectives function is called an optimal solution. Greedy is a strategy that works well on optimization problems with the following characteristics: • Greedy-choice property: A global optimum can be arrived at by selecting a local

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

> Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Design and Analysis of Algorithms module | PDF page 95 | lines 3-17

5. Introduction to Backtracking 5.1 Backtracking: • It is one of the most general algorithm design techniques. • Many problems which deal with searching for a set of solutions or for a optimal solution satisfying some constraints can be solved using the backtracking formulation. • To apply backtracking method, the desired solution must be expressible as an n- tuple (x1…xn) where xi is chosen from some finite set Si. • The problem is to find a vector, which maximizes or minimizes a criterion function P(x1….xn). • The major advantage of this method is, once we know that a partial vector (x1,…xi) will not lead to an optimal solution that (mi+1…….mn) possible test vectors may be ignored entirely. • Many problems solved using backtracking require that all the solutions satisfy a complex set of constraints.

Question sources

Computer Science MCQ practice collection · Question 371

Question 60 | AM0372

An exponential recursive optimization repeatedly solves the same well-defined states. Optimal substructure holds and there are only polynomially many distinct states with polynomial work per state. Which technique avoids the repeated work by storing state results?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: Dynamic programming

Dynamic programming stores the result for each distinct state, through memoization or tabulation, and reuses it instead of repeating equivalent recursive work. With the stated polynomial state count and polynomial work per state, this yields polynomial total work. Greedy needs an additional local-choice safety argument, divide and conquer targets independent instances, and plain backtracking does not provide the specified reuse.

Why each choice is right or wrong

A - Incorrect: Independent subproblem decomposition does not address the stated repeated states.

B - Incorrect: No safe irrevocable local-choice rule is supplied.

C - Correct: DP stores and reuses the repeated state results.

D - Incorrect: Plain candidate backtracking does not supply the requested cache reuse.

Primary note proof

Study-note section DAA25

Overlapping states motivate reuse; optimal substructure supports optimization. These are textbook signatures, not requirements that every DP problem optimize or every state have multiple users. DP also counts/decides. Cost=states×work per state and can remain exponential.

Study-note section DAA26

Memoization recursively checks a cache before solving a state and stores its result. Tabulation orders dependencies explicitly; chosen domains may exclude irrelevant states. Cache keys include all relevant parameters; recursion/locality/sparsity affect implementation cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Design and Analysis of Algorithms module | PDF page 48 | lines 17-22

Dynamic programming is a stage-wise search method suitable for optimization problems whose solutions may be viewed as the result of a sequence of decisions. The most attractive property of this strategy is that during the search for a solution it avoids full enumeration by pruning early partial decision solutions that cannot possibly lead to optimal solution. In many practical situations, this strategy hits the optimal solution in a polynomial number of decision steps. However, in the worst case, such a strategy may end up performing full enumeration.

Design and Analysis of Algorithms module | PDF page 95 | lines 3-17

5. Introduction to Backtracking 5.1 Backtracking: • It is one of the most general algorithm design techniques. • Many problems which deal with searching for a set of solutions or for a optimal solution satisfying some constraints can be solved using the backtracking formulation. • To apply backtracking method, the desired solution must be expressible as an n- tuple (x1…xn) where xi is chosen from some finite set Si. • The problem is to find a vector, which maximizes or minimizes a criterion function P(x1….xn). • The major advantage of this method is, once we know that a partial vector (x1,…xi) will not lead to an optimal solution that (mi+1…….mn) possible test vectors may be ignored entirely. • Many problems solved using backtracking require that all the solutions satisfy a complex set of constraints.

Question sources

Computer Science MCQ practice collection · Question 372

Question 61 | AM0373

Which technique combines solutions of overlapping subproblems using a defined state recurrence and reuses each solved state rather than solving it repeatedly?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: Dynamic programming

Dynamic programming defines states and a recurrence, solves dependencies in a valid order and reuses the resulting values. Combining those values forms larger solutions without repeating the same state work. Divide and conquer combines largely independent instances instead, greedy keeps safe local commitments, and backtracking grows and retracts candidates. The explicit overlap and reuse conditions make DP the specific match.

Why each choice is right or wrong

A - Incorrect: Divide and conquer emphasizes independent instances rather than reused overlapping states.

B - Incorrect: Greedy is defined by safe local commitments.

C - Correct: DP combines stored dependent state values.

D - Incorrect: Backtracking is defined by candidate search and retraction.

Primary note proof

Study-note section DAA25

> Define state meaning, choices, recurrence, bases and dependency order. Prove replacing a subsolution by a better one improves the full solution; otherwise state information may be missing. Show the recurrence covers all legal final choices and generates feasible candidates, then induct in dependency order. Store choices for reconstruction.

Study-note section DAA26

> Memoization recursively checks a cache before solving a state and stores its result. Tabulation orders dependencies explicitly; chosen domains may exclude irrelevant states. Cache keys include all relevant parameters; recursion/locality/sparsity affect implementation cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 17 | lines 5-9

> 2.1 Divide and Conquer Algorithm In this approach, we solve a problem recursively by applying 3 steps 1. DIVIDE-break the problem into several sub problems of smaller size. 2. CONQUER-solve the problem recursively. 3. COMBINE-combine these solutions to create a solution to the original problem.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

> Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Question sources

Computer Science MCQ practice collection · Question 373

Question 62 | AM0374

For nonnegative weighted edges, an algorithm repeatedly finalizes the unsettled vertex with minimum current source distance and relaxes its outgoing edges. Which design technique describes this Dijkstra procedure?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer B: Greedy algorithm

Dijkstra chooses the unsettled vertex of minimum distance and commits to its finalized value. Nonnegative edges ensure that a later path through other unsettled vertices cannot improve it. That safe irrevocable minimum choice is greedy. DP methods also solve other shortest-path settings, but they are not the explicitly described finalization procedure; independent decomposition and retracting path search are likewise different designs.

Why each choice is right or wrong

A - Incorrect: The procedure does not solve independent graph subinstances and combine them.

B - Correct: Safe minimum-distance finalization is a greedy choice.

C - Incorrect: A DP path recurrence is a different method from the specified Dijkstra procedure.

D - Incorrect: The algorithm finalizes distances rather than retracting path candidates.

Primary note proof

Study-note section DAA24

> Initialize dist[source]=0, others∞ and predecessors unset. Repeatedly extract the unsettled vertex u of minimum distance; relax each edge by dist[v]=min(dist[v], dist[u]+w(u, v)), updating predecessors when improved. Once the minimum is∞, remaining vertices are unreachable. A destination may be finalized before stopping, not merely discovered.

Study-note section DAA19

> A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 46 | lines 2-4

> o Single Source Shortest Path Algorithm (Dijkstra's) Single-source shortest path: Graphs can be used to represent the highway structure of a state or country with

Design and Analysis of Algorithms module | PDF page 61 | lines 14-23

> 4.2. ALL PAIR SHORTEST PATH • Let G=<V,E> be a directed graph 'V' is a set of nodes and _E' is the set of edges. • Each edge has an associated non-negative length. • We want to calculate the length of the shortest path between each pair of nodes. • Suppose the nodes of G are numbered from 1 to n, so V={1,2,...N},and suppose G matrix L gives the length of each edge, with L(i,j)=0 for i=1,2...n,L(i,j)>=for all i & j, and L(i,j)=infinity, if the edge (i,j) does not exist. • The principle of optimality applies: if k is the node on the shortest path from i to j then the part of the path from i to k and the part from k to j must also be optimal, that is shorter.

Design and Analysis of Algorithms module | PDF page 49 | lines 2-12

> The Principle of Optimality The dynamic programming relies on a principle of optimality. This principle states that in an optimal sequence of decisions or choices, each subsequence must also be optimal. For example, in matrix chain multiplication problem, not only the value we are interested in is optimal but all the other entries in the table are also represent optimal. The

principle can be related as follows: the optimal solution to a problem is a combination of optimal solutions to some of its sub problems. The difficulty in turning the principle of optimally into an algorithm is that it is not usually obvious which sub problems are relevant to the problem under consideration. General method-multistage graphs-all pair shortest path algorithm-Optimal Binary Search trees- 0/1 knapsack Problem-Reliability design-traveling salesman problem

Question sources

Computer Science MCQ practice collection · Question 374

Question 63 | AM0375

Which algorithm design technique is commonly used for sorting algorithms?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer A: Divide and conquer

Merge sort splits into independent halves and merges (PDF p.21), and quick sort partitions into independently sorted subarrays (p.23). These provide the concrete common sorting examples for divide and conquer. The answer is about those taught examples, not a claim that every sort uses this technique.

Why each choice is right or wrong

A - Correct: Both merge sort and quick sort are explicitly divide and conquer (pp.21,23).

B - Incorrect: No greedy-choice sorting construction is the taught example in pp.21-23.

C - Incorrect: No tabulated shared-subproblem sorting construction is presented in pp.21-23.

D - Incorrect: No candidate-solution backtracking sort is the taught example in pp.21-23.

Primary note proof

Study-note section DAA14

Split by indices, recursively sort each half, then repeatedly copy the smaller remaining front element into a temporary result. The merge invariant says the output prefix is sorted and contains the smallest elements already consumed. Append the unexhausted suffix. Equal keys must be taken from the left half first to preserve stability.

Study-note section DAA15

Choose a pivot, partition, then sort the resulting smaller ranges. In a two-pointer scheme, advance i while values are below pivot and j while above; when i≤j, swap and advance both. Bound the scans and ensure progress on equal values. Recurse on the resulting ranges. Hoare-style partition boundaries need not put the pivot at its final position; Lomuto-style partition commonly does. Do not mix their return conventions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 21 | lines 18-29

MERGE SORT • As another example divide-and-conquer, we investigate a sorting algorithm that has the nice property that is the worst case its complexity is O(n log n) • This algorithm is called merge sort • We assume throughout that the elements are to be sorted in non-decreasing order. • Given a sequence of _n' elements a[1],...,a[n] the general idea is to imagine then split into 2 sets a[1],.....,a[n/2] and a[[n/2]+1],....a[n]. • Each set is individually sorted, and the resulting sorted sequences are merged to produce a single sorted sequence of _n' elements. • Thus, we have another ideal example of the divide-and-conquer strategy in which the splitting is into 2 equal-sized sets & the combining operation is the merging of 2 sorted sets into one.

Design and Analysis of Algorithms module | PDF page 23 | lines 19-29

QUICK SORT • The divide-and-conquer approach can be used to arrive at an efficient sorting method different from merge sort. • In merge sort, the file a[1:n] was divided at its midpoint into sub arrays which were independently sorted & later merged. • In Quick sort, the division into 2 sub arrays is made so that the sorted sub arrays do not need to be merged later. • This is accomplished by rearranging the elements in a[1:n] such that a[I]<=a[j] for all I between 1 & n and all j between (m+1) & n for some m, 1<=m<=n. • Thus the elements in a[1:m] & a[m+1:n] can be independently sorted. • No merge is needed. This rearranging is referred to as partitioning.

Question sources

Computer Science MCQ practice collection · Question 375

Question 64 | AM0376

A sequence-decision problem has overlapping state subproblems and optimal substructure. Which technique evaluates all legal next choices through a recurrence and stores the best value for each state for reuse?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: Dynamic programming

Dynamic programming defines enough state to summarize the remaining decision problem, evaluates all legal choices and stores the best resulting value. Optimal substructure justifies using the stored optimal subsolutions, and overlap makes reuse useful. A mere decision sequence would not identify a technique; the explicit recurrence and reuse distinguish DP from greedy commitments, independent divide-and-conquer instances and backtracking without cached state values.

Why each choice is right or wrong

A - Incorrect: Independent decomposition does not describe the stated state reuse.

B - Incorrect: Greedy does not generally evaluate all choices through cached state recurrences.

C - Correct: DP evaluates and reuses best state values.

D - Incorrect: Backtracking without state caching revisits candidate choices rather than implementing this reuse rule.

Primary note proof

Study-note section DAA25

> Define state meaning, choices, recurrence, bases and dependency order. Prove replacing a subsolution by a better one improves the full solution; otherwise state information may be missing. Show the recurrence covers all legal final choices and generates feasible candidates, then induct in dependency order. Store choices for reconstruction.

Study-note section DAA26

> Memoization recursively checks a cache before solving a state and stores its result. Tabulation orders dependencies explicitly; chosen domains may exclude irrelevant states. Cache keys include all relevant parameters; recursion/locality/sparsity affect implementation cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 48 | lines 17-22

> Dynamic programming is a stage-wise search method suitable for optimization problems whose solutions may be viewed as the result of a sequence of decisions. The most attractive property of this strategy is that during the search for a solution it avoids full enumeration by pruning early partial decision solutions that cannot possibly lead to optimal solution. In many practical situations, this strategy hits the optimal solution in a polynomial number of decision steps. However, in the worst case, such a strategy may end up performing full enumeration.

Design and Analysis of Algorithms module | PDF page 49 | lines 27-31

> • The essential difference between the greedy method and dynamic programming is that the greedy method only one decision sequence is ever generated. • In dynamic programming, many decision sequences may be generated. However, sequences containing sub-optimal sub-sequences cannot be optimal and so will not be generated.

Design and Analysis of Algorithms module | PDF page 95 | lines 3-17

> 5. Introduction to Backtracking 5.1 Backtracking: • It is one of the most general algorithm design techniques. • Many problems which deal with searching for a set of solutions or for a optimal solution satisfying some constraints can be solved using the backtracking formulation. • To apply backtracking method, the desired solution must be expressible as an n- tuple (x1…xn) where xi is chosen from some finite set Si. • The problem is to find a vector, which maximizes or minimizes a criterion function P(x1….xn). • The major advantage of this method is, once we know that a partial vector (x1,…xi) will not lead to an optimal solution that (mi+1…….mn) possible test vectors may be ignored entirely. • Many problems solved using backtracking require that all the solutions satisfy a complex set of constraints.

Question sources

Computer Science MCQ practice collection · Question 376

Question 65 | AM0377

Which technique builds a candidate incrementally, checks constraints after each addition and undoes the last addition when it cannot lead to a complete feasible solution?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer D: Backtracking

Backtracking keeps a partial candidate, adds a choice, checks whether completion remains possible, and recursively explores promising extensions. On return it undoes the choice so other alternatives can be tried. Greedy also grows solutions but does not undo committed choices; DP builds reusable states rather than this candidate stack. Independent divide-and-conquer combination does not describe the specified constraint-search mechanism.

Why each choice is right or wrong

A - Incorrect: Independent subinstance combination is a different mechanism.

B - Incorrect: Greedy keeps rather than undoes its commitments.

C - Incorrect: DP is defined by state reuse rather than this candidate undo rule.

D - Correct: Constraint checks and undoing partial choices are backtracking.

Primary note proof

Study-note section DAA36

Represent a partial assignment x[1.. k]. For each allowed next choice: assign it, test whether an extension remains possible, recurse if promising, and undo the assignment on return. Report complete feasible candidates; stop after one only if the task requests existence/a single answer. Explicit constraints restrict each variable’s domain; implicit constraints relate choices.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 29 | lines 5-13

Algorithm Greedy (a,n) //a[1:n] contain the n inputs { solution =0;//Initialise the solution. For i=1 to n do { x=select(a); if(feasible(solution,x))then solution=union(solution,x); } return solution;

Design and Analysis of Algorithms module | PDF page 96 | lines 3-21

Algorithm Backtracking (n) // This schema describes the backtracking procedure. All solutions are generated in X[1:n] //and printed as soon as they are determined. { k=1; While (k ≠ 0) do { if (there remains all untried X[k] ∈ T (X[1],[2],.....X[k-1]) and Bk (X[1],.....X[k])) is true) then { if(X[1],.....X[k])is the path to the answer node) Then write(X[1:k]); k=k+1; //consider the next step. } else k=k-1; //consider backtracking to the previous set. }} • All solutions are generated in X[1:n] and printed as soon as they are determined. • T(X[1].....X[k-1]) is all possible values of X[k] gives that X[1],.....X[k-1] have already been chosen.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Question sources

Computer Science MCQ practice collection · Question 377

Question 66 | AM0379

Which technique finds maximum contiguous subarray by recursively comparing the best left-half, right-half and crossing subarray, with a linear-time crossing computation?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer A: Divide and conquer

The stated construction splits the array, recursively solves each half and combines those results with the best crossing interval. This is divide and conquer, with recurrence $2T(n/2)+\Theta(n)=\Theta(n \log n)$. Kadane also solves the same problem, but uses a best-sum-ending-at-index DP recurrence rather than this construction. Greedy commitments and extend/retract search do not describe the specified three-way comparison.

Why each choice is right or wrong

A - Correct: Independent halves plus the crossing combination define divide and conquer.

B - Incorrect: No irrevocable local-choice rule defines this construction.

C - Incorrect: Kadane is DP, but it is not the described left/right/crossing algorithm.

D - Incorrect: The construction does not extend and retract candidate assignments.

Primary note proof

Study-note section DAA18

> A maximum subarray is contiguous; it is not an arbitrary subsequence. Divide-and-conquer compares the best left, best right and best crossing result. The crossing result is the best suffix of the left half plus best prefix of the right, found in $\Theta(n)$. This yields $2T(n/2)+\Theta(n)=\Theta(n \log n)$.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 17 | lines 5-9

> 2.1 Divide and Conquer Algorithm In this approach, we solve a problem recursively by applying 3 steps 1. DIVIDE-break the problem into several sub problems of smaller size. 2. CONQUER-solve the problem recursively. 3. COMBINE-combine these solutions to create a solution to the original problem.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Question sources

Computer Science MCQ practice collection · Question 379

Question 67 | AM0380

Which listed techniques describe exact TSP algorithms using either (i) cached subset-and-endpoint states or (ii) complete extend-and-retract enumeration of feasible tours while retaining the cheapest? Select all that apply.

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C, D: Dynamic programming; Backtracking

Held-Karp stores the best cost for a visited subset and endpoint, which is dynamic programming. Complete backtracking extends feasible tours, retracts choices and retains the least-cost complete tour. Both are exact constructions described by the stem. Greedy nearest-neighbor can produce a tour but has no general optimality guarantee, while independent divide-and-conquer combination is not either specified construction.

Why each choice is right or wrong

A - Incorrect: Neither specified construction is independent divide-and-conquer combination.

B - Incorrect: A greedy tour heuristic is not a general exact optimizer.

C - Correct: Cached subset-and-endpoint states form Held-Karp DP.

D - Correct: Complete extend-and-retract tour search is backtracking.

Primary note proof

Study-note section DAA33

Let F(S, j) be minimum cost from s through exactly S, ending at j∈S.

Study-note section DAA33

Held-Karp uses O(V²2^V) time and O(V 2^V) space, improving factorial enumeration but remaining exponential. Greedy nearest-neighbor is a heuristic without a general exactness guarantee; backtracking and branch and bound can also solve exact TSP.

Study-note section DAA36

For each allowed next choice: assign it, test whether an extension remains possible, recurse if promising, and undo the assignment on return.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 78 | lines 6-13

4.6. TRAVELLING SALESMAN PROBLEM Let G(V,E) be a directed graph with edge cost cij is defined such that cij >0 for all i and j and Cij =∞ ,if <i,j> € E. Let |V|=n and assume n>1. The traveling salesman problem is to find a tour of minimum cost. A tour of G is a directed cycle that include every vertex in V. The cost of the tour is the sum of cost of the edges on the tour. The tour is the shortest path that starts and ends at the same vertex i.e., 1.

Design and Analysis of Algorithms module | PDF page 49 | lines 2-12

The Principle of Optimality The dynamic programming relies on a principle of optimality. This principle states that in an optimal sequence of decisions or choices, each subsequence must also be optimal. For example, in matrix chain multiplication problem, not only the value we are interested in is optimal but all the other entries in the table are also represent optimal. The principle can be related as follows: the optimal solution to a problem is a combination of optimal solutions to some of its sub problems. The difficulty in turning the principle of optimally into an algorithm is that it is not usually obvious which sub problems are relevant to the problem under consideration. General method-multistage graphs-all pair shortest path algorithm-Optimal Binary Search trees- 0/1 knapsack Problem-Reliability design-traveling salesman problem

Design and Analysis of Algorithms module | PDF page 30 | lines 10-14

Greedy algorithms sometimes fail to produce the optimal solution, and may even produce the unique worst possible solution. One example is the travelling salesman problem. Greedy algorithms can be characterized as being 'short sighted', and as 'non-recoverable'. They are ideal only for problems which have 'optimal substructure'. Despite this, greedy algorithms are best suited for simple problems (e.g. giving change). It is important, however, to note that

Design and Analysis of Algorithms module | PDF page 130 | lines 18-23

TRAVELLING SALESMAN PROBLEM INTRODUCTION: It is algorithmic procedures similar to backtracking in which a new branch is chosen and is there (bound there) until new branch is choosing for advancing. Travelling salesman is a minimization problem. It can be solved using Branch and bound by generating state space tree by calculating cost and upper bound.

Question sources

Computer Science MCQ practice collection · Question 380

Question 68 | AM0381

Which algorithm design technique is used for solving problems that can be solved by making the best local choice at each step?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer B: Greedy algorithm

The greedy-choice property selects a local optimum, and the method uses immediately available information (PDF p.27). The stem's best local choice identifies greedy. PDF p.30 warns that this is not automatically a global optimality proof.

Why each choice is right or wrong

A - Incorrect: Independent splitting is not the p.27 local-choice criterion.

B - Correct: Greedy directly matches the p.27 short-sighted local-selection description.

C - Incorrect: Dynamic programming compares subproblem solutions rather than committing solely to the best current choice (p.30 contrast).

D - Incorrect: Backtracking revisits choices rather than making the non-recoverable choices described on p.30.

Primary note proof

Study-note section DAA19

> A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it. Optimal substructure alone does not prove this safe. An exchange proof inserts the greedy choice into an optimum without worsening cost; recurse. MST cut proofs use this exchange.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 27 | lines 7-18

> • Greedy method is the most straightforward designed technique. • As the name suggest they are short sighted in their approach taking decision on the basis of the information immediately at the hand without worrying about the effect these decisions may have in the future. DEFINITION:

• A problem with N inputs will have some constraints. Any subsets that satisfy these constraints are called a feasible solution. • A feasible solution that either maximize can minimize a given objectives function is called an optimal solution. Greedy is a strategy that works well on optimization problems with the following characteristics: • Greedy-choice property: A global optimum can be arrived at by selecting a local

Design and Analysis of Algorithms module | PDF page 30 | lines 10-14

Greedy algorithms sometimes fail to produce the optimal solution, and may even produce the unique worst possible solution. One example is the travelling salesman problem. Greedy algorithms can be characterized as being 'short sighted', and as 'non-recoverable'. They are ideal only for problems which have 'optimal substructure'. Despite this, greedy algorithms are best suited for simple problems (e.g. giving change). It is important, however, to note that

Question sources

Computer Science MCQ practice collection · Question 381

Question 69 | AM0382

Which algorithm design technique is used for finding the longest common subsequence between two sequences?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: Dynamic programming using prefix-pair states.

DAA30 defines a state for each pair of prefix lengths. A match extends a diagonal state; a mismatch takes the larger of two neighboring prefix solutions. Filling states in dependency order avoids recomputing overlapping subproblems and gives $\Theta(mn)$ time. This is the note's dynamic-programming solution; no greedy safety rule or backtracking enumeration is supplied.

Why each choice is right or wrong

A - Incorrect: The given LCS construction reuses overlapping prefix states rather than only independently solving and combining disjoint pieces.

B - Incorrect: No rule for committing to a locally best character is justified; prefix alternatives must be compared.

C - Correct: The prefix-pair table and recurrence reuse optimal subproblem lengths, exactly the dynamic-programming technique described.

D - Incorrect: The note computes stored prefix states instead of exhaustively extending and undoing candidate subsequences.

Primary note proof

Study-note section DAA30

A subsequence preserves order but need not be contiguous. For prefixes of lengths i, j define L[i, j]. Empty-prefix rows/columns are 0. If X[i−1]=Y[j−1], set L[i, j]=1+L[i−1, j−1]; otherwise use max(L[i−1, j], L[i, j−1]). Keep prefix lengths i and j distinct from the zero-based character indices i−1 and j−1.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Design and Analysis of Algorithms module | PDF page 49 | lines 2-12

The Principle of Optimality The dynamic programming relies on a principle of optimality. This principle states that in an optimal sequence of decisions or choices, each subsequence must also be optimal. For example, in matrix chain multiplication problem, not only the value we are interested in is optimal but all the other entries in the table are also represent optimal. The principle can be related as follows: the optimal solution to a problem is a combination of optimal solutions to some of its sub problems. The difficulty in turning the principle of optimally into an algorithm is that it is not usually obvious which sub problems are relevant to the problem under consideration. General method-multistage graphs-all pair shortest path algorithm-Optimal Binary Search trees- 0/1 knapsack Problem-Reliability design-traveling salesman problem

Question sources

Computer Science MCQ practice collection · Question 382

Question 70 | AM0383

Which algorithm design technique is used for solving problems that can be divided into smaller independent subproblems?

A. Divide and conquer

B. Greedy algorithm

C. Dynamic programming

D. Backtracking

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer A: Divide and conquer

PDF p.48 explicitly describes divide and conquer as partitioning into independent subproblems, solving recursively, and combining solutions. The independence qualification makes this stem more specific than merely saying “subproblems.”

Why each choice is right or wrong

A - Correct: A directly matches independent partitioning and recursive solution (pp.17,48).

B - Incorrect: Greedy is distinguished by local feasible selection, not independent recursive partitioning (p.48 contrast).

C - Incorrect: Dynamic programming is motivated by subproblems sharing sub-subproblems (p.48).

D - Incorrect: Backtracking searches constrained assignments rather than the independent partition framework (p.48).

Primary note proof

Study-note section DAA12

> Divide and conquer solves smaller largely independent instances and combines results. DP reuses a defined set of subproblem states. Greedy commits to a feasible local choice only when a safety argument supports it. Backtracking grows/retracts candidates when constraints fail; branch and bound also rejects candidates whose best possible objective cannot beat an incumbent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 17 | lines 5-9

> 2.1 Divide and Conquer Algorithm In this approach, we solve a problem recursively by applying 3 steps 1. DIVIDE-break the problem into several sub problems of smaller size. 2. CONQUER-solve the problem recursively. 3. COMBINE-combine these solutions to create a solution to the original problem.

Design and Analysis of Algorithms module | PDF page 48 | lines 5-16

> Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapter 2 of binary search, divide-and-conquer algorithms partition the problem into independent subproblems, solve the subproblems recursively, and then combine their solutions to solve the original problem. In contrast, dynamic programming is applicable when the subproblems are not independent, that is, when subproblems share sub subproblems. In this context, a divide-and-conquer algorithm does more work than necessary, repeatedly solving the common sub subproblems. A dynamic-programming algorithm solves every subproblem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the subproblem is encountered. Dynamic programming is typically applied to optimization problems. In such problems there can be many possible solutions. Each solution has a value, and we wish to find a solution with the optimal (minimum or maximum) value.

Question sources

Computer Science MCQ practice collection · Question 383

Question 71 | AM0384

Which term refers to the amount of resources, such as time and memory, required by an algorithm to solve a problem?

A. Efficiency

B. Complexity

C. Optimality

D. Completeness

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer B: Complexity

Complexity expresses algorithm requirements in time and space (PDF p.6), with performance defined as needed computer memory and time (p.4). Efficiency is an evaluation of those requirements; complexity names the resource requirement being quantified.

Why each choice is right or wrong

A - Incorrect: Efficiency evaluates how economically resources are used, while p.6 calls their expression complexity.

B - Correct: Complexity is explicitly expressed in time and space requirements (p.6).

C - Incorrect: Optimality concerns the objective value of a solution, not the resources used to find it (p.6 contrast).

D - Incorrect: Completeness concerns fully solving a problem, not its memory/time quantity (p.4 context).

Primary note proof

Study-note section DAA02

Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Design and Analysis of Algorithms module | PDF page 4 | lines 6-19

• Finiteness: Algorithm must complete after a finite number of steps. • Definiteness: Each step must be clearly defined, having one and only one interpretation. At each point in computation, one should be able to tell exactly what happens next. • Sequence: Each step must have a unique defined preceding and succeeding step. The first step (start step) and last step (halt step) must be clearly noted. • Feasibility: It must be possible to perform each instruction. • Correctness: It must compute correct answer for all possible legal inputs. • Language Independence: It must not depend on any one programming language. • Completeness: It must solve the problem completely. • Effectiveness: It must be possible to perform each step exactly and in a finite amount of time. • Efficiency: It must solve with the least amount of computational resources such as time and space. • Generality: Algorithm should be valid on all possible inputs • Input/Output: There must be a specified number of input values, and one or more result values.

Question sources

Computer Science MCQ practice collection · Question 384

Question 72 | AM0385

What is the purpose of analyzing the computational complexity of algorithms?

A. To determine the correctness of an algorithm

B. To measure the size of the input data

C. To evaluate the performance of an algorithm

D. To compare different programming languages

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: To evaluate the performance of an algorithm

Analysis estimates memory and time needed to solve an input problem (PDF p.4); these estimates help select between possible algorithms (p.6). Evaluating performance is therefore the direct purpose.

Why each choice is right or wrong

A - Incorrect: Correctness concerns whether the result is right; the p.4 performance definition concerns resources.

B - Incorrect: Input size is an argument of the analysis, not the ultimate performance-evaluation purpose (pp.4,6).

C - Correct: Evaluating resource performance directly matches pp.4,6.

D - Incorrect: The comparison is between algorithms and their requirements, not primarily programming languages (p.6).

Primary note proof

Study-note section DAA02

> Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 20-28

> 1.3. Complexity analysis 1.3.1. Space and Time Complexity Performance Analysis By performance of a program, we mean the amount of computer memory and time needed to run a program. There are two approaches to determine the performance of a program: - Analytical approach- used for performance analysis, - Experimental approach - conduct experiments for performance

measurement. What is Analysis: - The analysis of algorithm provides background information that a general idea of how long an algorithm will take for a given set. For each algorithm considered, we will come up with an estimate of how

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Design and Analysis of Algorithms module | PDF page 4 | lines 6-19

• Finiteness: Algorithm must complete after a finite number of steps. • Definiteness: Each step must be clearly defined, having one and only one interpretation. At each point in computation, one should be able to tell exactly what happens next. • Sequence: Each step must have a unique defined preceding and succeeding step. The first step (start step) and last step (halt step) must be clearly noted. • Feasibility: It must be possible to perform each instruction. • Correctness: It must compute correct answer for all possible legal inputs. • Language Independence: It must not depend on any one programming language. • Completeness: It must solve the problem completely. • Effectiveness: It must be possible to perform each step exactly and in a finite amount of time. • Efficiency: It must solve with the least amount of computational resources such as time and space. • Generality: Algorithm should be valid on all possible inputs • Input/Output: There must be a specified number of input values, and one or more result values.

Question sources

Computer Science MCQ practice collection · Question 385

Question 73 | AM0387

What does f(n)∈O(g(n)) mean, independently of whether f measures best, average, worst or space cost?

A. The best-case time complexity of an algorithm

B. The average-case time complexity of an algorithm

C. An eventual constant-factor upper bound: f(n)≤c·g(n) for fixed c>0 and all sufficiently large n.

D. The space complexity of an algorithm

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: An eventual constant-factor upper bound: f(n)≤c·g(n) for fixed c>0 and all sufficiently large n.

Big-O gives an eventual constant-factor upper bound: for some fixed c>0 and n_0, f(n)≤c·g(n) whenever n≥n_0. It does not choose best, average or worst inputs, nor does it specify time rather than space. Those choices determine the function being bounded. Option C states the general mathematical relation that applies across all of those analyses.

Why each choice is right or wrong

A - Incorrect: Best case is a way to define f, not the meaning of Big-O.

B - Incorrect: Average case defines an expectation, not the bound relation.

C - Correct: The eventual upper inequality is the definition of Big-O.

D - Incorrect: Space is one possible resource for f, not the definition of the notation.

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, f∈O(g) means f(n)≤c·g(n) for all n≥n 0, with fixed c>0. Ω reverses that inequality; Θ requires both bounds. Little-o means f/g→0, and little-ω means f/g→∞ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Study-note section DAA03

> At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

> In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the ―Big-Oh‖ estimate.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

> I. Big-Oh Notation (O): It’s only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Data Structure and Algorithms module | PDF page 3 | lines 6-12

II. Big-Omega Notation (Ω): provides an asymptotic lower bound; f(n)= Ω(g (n)) means that the growth rate of f(n) is greater than or equal to g(n). III. Theta Notation (Θ): If f(n)= Θ (g(n)), then g(n) is an asymptotically tight bound for f(n) i.e f(n) and g(n) have the same rate of growth. IV. Little-o Notation (o): f(n)=o(g(n)) means f(n) has less growth rate compared to g(n). V. Little-Omega Notation (ω): ω notation used to denote a lower bound that is not asymptotically tight.

Question sources

Computer Science MCQ practice collection · Question 387

Question 74 | AM0388

Which time complexity indicates an algorithm that executes in constant time regardless of the input size?

A. O(1)

B. O(n)

C. O(log n)

D. O(n^2)

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer A: O(1)

A bound independent of input size is constant O(1). This is the constant K in the visually inspected related-module p.3 growth diagram and agrees with classifying growth rather than exact operation count on assigned PDF p.14.

Why each choice is right or wrong

A - Correct: 1 is independent of n and gives the constant upper bound (cross-module p.3).

B - Incorrect: n grows as the input increases, unlike a constant (assigned p.14).

C - Incorrect: log n still grows, though slowly (assigned p.14).

D - Incorrect: n^2 increases quadratically rather than staying fixed (assigned p.14).

Primary note proof

Study-note section DAA04

For eventually nonnegative f, g, f∈O(g) means f(n)≤c·g(n) for all n≥n 0, with fixed c>0. Ω reverses that inequality; Θ requires both bounds. Little-o means f/g→0, and little-ω means f/g→∞ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Study-note section DAA05

Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Question sources

Computer Science MCQ practice collection · Question 388

Question 75 | AM0389

Which term describes the amount of memory required by an algorithm to solve a problem?

A. Time complexity

B. Space complexity

C. Computational complexity

D. Algorithmic complexity

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer B: Space complexity

PDF p.6 explicitly defines space complexity as the amount of memory space needed to run a program to completion. B is the specific resource name, while the other complexity terms are broader or refer to duration.

Why each choice is right or wrong

A - Incorrect: Time complexity measures time rather than memory (p.6).

B - Correct: Space complexity is the precise definition on p.6.

C - Incorrect: Computational complexity is broader than the memory-specific measure asked for (p.6 distinguishes resources).

D - Incorrect: Algorithmic complexity is broader; p.6 names the particular memory measure space complexity.

Primary note proof

Study-note section DAA08

> Space complexity measures peak simultaneously live storage, not the sum of every allocation ever made. Count input, auxiliary structures and stack according to the requested convention. n active recursion levels with constant-size frames need Θ(n) stack; depth log n with constant frames needs Θ(log n). The number of total calls does not determine peak depth.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 6 | lines 4-10

> The complexity of an algorithm can be expressed in terms of its space and time complexity. Space Complexity of a program: - is the amount of memory space it needs to run to completion. We are interested in the space complexity of a program for the following reasons: • If a program is to run on a multiuser computer system, then we may need to specify the amount of memory to be allocated for the program. • For any computer system, we would like to know in advance whether or not sufficient memory is available to run the program.

Design and Analysis of Algorithms module | PDF page 4 | lines 6-19

> • Finiteness: Algorithm must complete after a finite number of steps. • Definiteness: Each step must be clearly defined, having one and only one interpretation. At each point in computation, one should be able to tell exactly what happens next. • Sequence: Each step must have a unique defined preceding and succeeding step. The first step (start step) and last step (halt step) must be clearly noted. • Feasibility: It must be possible to perform each instruction. • Correctness: It must compute correct answer for all possible legal inputs. • Language Independence: It must not depend on any one programming language. • Completeness: It must solve the problem completely. • Effectiveness: It must be possible to perform each step exactly and in a finite amount of time. • Efficiency: It must solve with the least amount of computational resources such as time and space. • Generality: Algorithm should be valid on all possible inputs • Input/Output: There must be a specified number of input values, and one or more result values.

Question sources

Computer Science MCQ practice collection · Question 389

Question 76 | AM0390

What does the term "worst-case analysis" refer to in computational complexity?

A. Analyzing the best-case performance of an algorithm

B. Analyzing the average-case performance of an algorithm

C. Analyzing the worst possible input for an algorithm

D. Analyzing the expected performance of an algorithm

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: Analyzing the worst possible input for an algorithm

Worst-case analysis considers the worst possible input among inputs of a fixed size, giving a bound covering all their runtimes (PDF p.14). C describes that selection; average and expected costs answer different questions.

Why each choice is right or wrong

A - Incorrect: Best case considers the fastest rather than slowest input (p.14).

B - Incorrect: Average case uses an average input rather than the maximizing one (p.14).

C - Correct: The worst possible input is precisely the p.14 definition.

D - Incorrect: Expected performance requires averaging rather than selecting the slowest input (p.14).

Primary note proof

Study-note section DAA03

> At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 3-9

> In order to determine the running time of an algorithm it is possible to define three functions Tbest(n), Tavg(n) and Tworst(n) as the best, the average and the worst case running time of the algorithm respectively. Average Case (Tavg): The amount of time the algorithm takes on an "average" set of inputs. Worst Case (Tworst): The amount of time the algorithm takes on the worst possible set of inputs. Best Case (Tbest): The amount of time the algorithm takes on the

> smallest possible set of inputs. We are interested in the worst-case time, since it provides a bound for all input - this is called the —Big-Oh‖ estimate.

Question sources

Computer Science MCQ practice collection · Question 390

Question 77 | AM0391

What is the significance of logarithmic time complexity, such as O(log n)?

A. It indicates linear time complexity.

B. It indicates constant time complexity.

C. It indicates exponential time complexity.

D. It indicates sublinear time complexity.

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer D: It indicates sublinear time complexity.

For increasing positive n, log n grows more slowly than n: repeated halving reaches unit size in only log_2 n stages, as illustrated by binary search (Data Structures and Algorithms PDF p.3). Assigned PDF p.14 separately lists logarithmic and linear classes. Therefore logarithmic growth is sublinear.

Why each choice is right or wrong

A - Incorrect: Linear is proportional to n, faster than log n (cross-module p.3 search comparison).

B - Incorrect: Logarithms remain unbounded, so they are not constant (assigned p.14).

C - Incorrect: Exponential growth is represented by 2^n, not log n (assigned p.14).

D - Correct: Log n grows slower than n, so it is sublinear (cross-module p.3 search comparison).

Primary note proof

Study-note section DAA05

> Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the

size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Data Structure and Algorithms module | PDF page 3 | lines 23-31

Binary Searching • Works on ordered/sorted list • Uses divide and conquer method: it locates mid-point and identify which have the target belongs to since the list is sorted. It performs this recursively till the item found or the list is empty(target is not in the list • The computational time for this algorithm is proportional to log2 n.

Data Structure and Algorithms module | PDF page 3 | lines 16-22

Linear/Sequential searching • Works with unsorted List • Loop through the list starting at the first element • Time is proportional to the size of input (n) and we call this time complexity O(n).

Question sources

Computer Science MCQ practice collection · Question 391

Question 78 | AM0392

What does the term "P versus NP problem" refer to in computational complexity theory?

A. The problem of distinguishing between parallel and non-parallel algorithms.

B. The problem of determining the runtime of algorithms.

C. The problem of comparing the efficiency of programming languages.

D. The problem of determining if certain types of problems have efficient algorithms.

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer D: Whether efficiently verifiable decision problems also have efficient solving algorithms.

DAA44 defines P by deterministic polynomial-time decision solving and NP by polynomial-time certificate verification, equivalently nondeterministic polynomial solving. The P-versus-NP problem asks whether these classes coincide. Option D captures the question of efficient algorithms for this particular class of problems, although its wording is broad. It does not concern parallelism, language comparisons, or timing one program.

Why each choice is right or wrong

A - Incorrect: P and NP refer to polynomial solving and verification classes, not parallel versus non-parallel execution.

B - Incorrect: The question concerns equality of decision-problem classes, not merely determining a particular algorithm runtime.

C - Incorrect: Programming-language efficiency is unrelated to whether polynomial-time solvability and verifiability coincide.

D - Correct: This broadly expresses whether NP problems have deterministic polynomial-time solving algorithms, the P-versus-NP question.

Primary note proof

Study-note section DAA44

> P contains decision problems solvable deterministically in polynomial input-bit time. NP contains decision problems with polynomial-length certificates verifiable in polynomial time, equivalently solvable by a nondeterministic polynomial-time machine. NP does not mean non-polynomial; P⊆NP. Whether P=NP remains open.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 30 | lines 10-14

> Greedy algorithms sometimes fail to produce the optimal solution, and may even produce the unique worst possible solution. One example is the travelling salesman problem. Greedy algorithms can be characterized as being 'short sighted', and as 'non-recoverable'. They are ideal only for problems which have 'optimal substructure'. Despite this, greedy algorithms are best suited for simple problems (e.g. giving change). It is important, however, to note that

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

> In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Question sources

Computer Science MCQ practice collection · Question 392

Question 79 | AM0393

Which complexity class includes problems that can be solved by a non-deterministic Turing machine in polynomial time?

A. P

B. NP

C. NP-complete

D. Exponential

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer B: NP

DAA44 identifies NP with decision problems solvable by a nondeterministic polynomial-time machine. P is a contained class defined using deterministic machines; NP-complete adds hardness, which the stem never requires. Exponential is a time-growth description rather than the requested definition. The alternate set replaces Exponential with NP-hard, which likewise does not change the defining NP answer.

Why each choice is right or wrong

A - Incorrect: P is defined through deterministic polynomial time; it is contained in NP but does not name the full defined class.

B - Correct: NP exactly names decision problems solvable by a nondeterministic polynomial-time Turing machine.

C - Incorrect: NP-complete requires both NP membership and NP-hardness; hardness is absent from the stem.

D - Incorrect: Exponential describes time growth, not the class defined by nondeterministic polynomial-time computation.

Primary note proof

Study-note section DAA44

> P contains decision problems solvable deterministically in polynomial input-bit time. NP contains decision problems with polynomial-length certificates verifiable in polynomial time, equivalently solvable by a nondeterministic polynomial-time machine. NP does not mean non-polynomial; P⊆NP. Whether P=NP remains open.

Study-note section DAA44

> A polynomial many-one reduction A≤p B transforms each A-instance into a B-instance preserving yes/no. To prove B NP-hard, reduce a known hard A to B, not the reverse.
> NP-complete means NP-hard and in NP; optimization TSP is normally called NP-hard, while its bounded-tour decision form is NP-complete. A valid proposed tour and its cost are easy to check,

but certifying global optimality is another task. Pseudo-polynomial O(nW) knapsack is not polynomial in the bit length of W.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 30 | lines 10-14

Greedy algorithms sometimes fail to produce the optimal solution, and may even produce the unique worst possible solution. One example is the travelling salesman problem. Greedy algorithms can be characterized as being 'short sighted', and as 'non-recoverable'. They are ideal only for problems which have 'optimal substructure'. Despite this, greedy algorithms are best suited for simple problems (e.g. giving change). It is important, however, to note that

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Variant 1 | Alternative version 1

Which complexity class includes problems that can be solved by a non-deterministic Turing machine in polynomial time?

A. P

B. NP

C. NP-complete

D. NP-hard

Answer B: NP

The variant retains P, NP, and NP-complete and replaces the fourth option with NP-hard. DAA44 still defines NP by nondeterministic polynomial-time computation. NP-hard is a reduction-based hardness property and need not imply NP membership, so the answer remains NP after independently examining this set.

Why each choice is right or wrong

A - Incorrect: P specifies deterministic polynomial solving, a subclass rather than the full nondeterministic polynomial class.

B - Correct: NP is exactly the class defined by nondeterministic polynomial-time decision computation.

C - Incorrect: NP-complete additionally requires NP-hardness; the stem imposes no such condition.

D - Incorrect: NP-hardness alone does not imply membership in NP or a nondeterministic polynomial-time algorithm.

Primary note proof

Study-note section DAA44

> P contains decision problems solvable deterministically in polynomial input-bit time. NP contains decision problems with polynomial-length certificates verifiable in polynomial time, equivalently solvable by a nondeterministic polynomial-time machine. NP does not mean non-polynomial; P⊆NP. Whether P=NP remains open.

Study-note section DAA44

> A polynomial many-one reduction A≤p B transforms each A-instance into a B-instance preserving yes/no. To prove B NP-hard, reduce a known hard A to B, not the reverse. NP-complete means NP-hard and in NP; optimization TSP is normally called NP-hard, while its bounded-tour decision form is NP-complete. A valid proposed tour and its cost are easy to check, but certifying global optimality is another task. Pseudo-polynomial O(nW) knapsack is not polynomial in the bit length of W.

Question sources

Computer Science MCQ practice collection · Question 393

Computer Science MCQ practice collection · Question 750

Question 80 | CS0114

For p=10 and n=5, execute if(n>p){print(n+p);}. Treat the comparison, branch and possible output as executable constant-cost operations. Which assessment is correct without adding an unstated output requirement?

A. Feasibility is violated.

B. Finiteness is violated.

C. Sequential control flow is violated.

D. No violation is demonstrated by this fragment.

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer D: No violation is demonstrated by this fragment.

The condition 5>10 is false, so the print is skipped and the fragment terminates after constant work. Its operations are executable and its conditional control flow is legitimate. Nothing in this specified fragment violates feasibility or finiteness, and an unrequired output is not a correctness obligation. Thus the accurate assessment is that no property violation is demonstrated by this fragment.

Why each choice is right or wrong

A - Incorrect: The specified operations are executable, so feasibility is satisfied.

B - Incorrect: The fragment terminates, so finiteness is satisfied.

C - Incorrect: A conditional branch is legitimate algorithmic control flow.

D - Correct: The false branch skips output and terminates without the claimed property violations.

Primary note proof

Study-note section DAA45

For p=10,n=5, if(n>p) cout<<(n+p) skips output and terminates. Skipping this output does not by itself violate feasibility, finiteness or control flow. Efficiency or a missing required output needs a specification.

Study-note section DAA01

State inputs, output, constraints and empty-solution policy. Algorithms are finite, definite executable methods independent of source language; feasible satisfies constraints, optimal is best feasible. Conditional branches are legitimate control flow.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 4 | lines 6-19

• Finiteness: Algorithm must complete after a finite number of steps. • Definiteness: Each step must be clearly defined, having one and only one interpretation. At each point in computation, one should be able to tell exactly what happens next. • Sequence: Each step must have a unique defined preceding and succeeding step. The first step (start step) and last step (halt step) must be clearly noted. • Feasibility: It must be possible to perform each instruction. • Correctness: It must compute correct answer for all possible legal inputs. • Language Independence: It must not depend on any one programming language. • Completeness: It must solve the problem completely. • Effectiveness: It must be possible to perform each step exactly and in a finite amount of time. • Efficiency: It must solve with the least amount of computational resources such as time and space. • Generality: Algorithm should be valid on all possible inputs • Input/Output: There must be a specified number of input values, and one or more result values.

Question sources

Computer Science Exit Exam practice collection · Question 114

Question 81 | CS0115

Assume it is given that f(n) = 4n^2, g(n) = 8n. Which asymptotic notation is correct?

A. f(n) = Big-Oh(g(n))

B. f(n) = Little-Oh(g(n))

C. f(n) = Big-Omega(g(n))

D. f(n) = Theta(g(n))

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: f(n) = Big-Omega(g(n))

Compare f/g=(4n^2)/(8n)=n/2, which grows without bound. Thus f eventually exceeds a constant multiple of g, establishing Omega(g) and excluding O(g), little-o(g) and Theta(g). The assigned PDF p.14 introduces growth comparison; the precise upper/lower/tight definitions are in Data Structures and Algorithms PDF p.3. For example n>=2 gives f>=g.

Why each choice is right or wrong

A - Incorrect: O(g) requires an eventual constant upper multiple; n/2 is unbounded, so it fails (cross-module p.3).

B - Incorrect: Little-o(g) requires slower growth; the ratio instead grows without bound (cross-module p.3).

C - Correct: Omega(g) is a lower growth bound; f>=g for every n>=2 establishes it (cross-module p.3).

D - Incorrect: Theta(g) requires the same growth rate; quadratic and linear differ by unbounded n/2 (cross-module p.3).

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, $f \in O(g)$ means $f(n) \le c \cdot g(n)$ for all $n \ge n_0$, with fixed $c>0$. Ω reverses that inequality; Θ requires both bounds. Little-o means $f/g \to 0$, and little-ω means $f/g \to \infty$ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Study-note section DAA05

> For $f=4n^2$ and $g=8n$, $f/g=n/2 \to \infty$. Therefore $f \in \Omega(g)$ and even $\omega(g)$, while f is neither $O(g)$ nor $\Theta(g)$.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 14 | lines 11-21

In analysis of algorithms, it is not important to know how many operations an algorithm does. Of greater concern is the rate of increase in operations for an algorithm to solve a problem as the size of the problem increases. This is referred to us the rate of growth of the algorithm. What happens with small sets of input data is not as interesting as what happens when the data set gets large. The rate of growth of an algorithm is dominated by the largest term in an equation; we will discard the terms that grow more slowly. When we strip all of these things away, we are left with what we call the order of the function or related algorithm. We then group algorithms based on their order. We group in three categories - those that grow at least as fast as some function, those that grow at the same rate, and those that grow no faster. The following are some of the classes of algorithms: - log2 n, n, n ∗ log2 n, n2,n3, 2n. Asymptotic analysis is concerned with how the running time of an algorithm increases with the size of the input in the limit, as the size of the input increases without bound.

Data Structure and Algorithms module | PDF page 3 | lines 2-5

I. Big-Oh Notation (O): It's only concerned with what happens for very a large value of n. Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n. An upper bound is the best algorithmic solution that has been found for a problem. NB: Each of the following functions is big-O of its successors:

Data Structure and Algorithms module | PDF page 3 | lines 6-12

II. Big-Omega Notation (Ω): provides an asymptotic lower bound; f(n)= Ω(g (n)) means that the growth rate of f(n) is greater than or equal to g(n). III. Theta Notation (Θ): If f(n)= Θ (g(n)), then g(n) is an asymptotically tight bound for f(n) i.e f(n) and g(n) have the same rate of growth. IV. Little-o Notation (o): f(n)=o(g(n)) means f(n) has less growth rate compared to g(n). V. Little-Omega Notation (ω): ω notation used to denote a lower bound that is not asymptotically tight.

Question sources

Computer Science Exit Exam practice collection · Question 115

Question 82 | CS0116

For a connected weighted undirected graph, which statements about its minimum spanning trees are true? Select all that apply; weights may tie.

A. Minimum spanning tree contains all vertices from an input graph

B. We can have only one unique minimum spanning tree for a graph

C. Minimum spanning trees are graphs

D. Minimum spanning trees form a cycle

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer A, C: Minimum spanning tree contains all vertices from an input graph; Minimum spanning trees are graphs

An MST contains every vertex and is a tree, hence is itself a graph. It is acyclic and has V−1 edges. Equal edge weights may permit multiple different MSTs, so uniqueness is not guaranteed. These properties make A and C true, while B's universal uniqueness claim and D's cycle claim are false. Distinct edge weights would be a sufficient additional uniqueness condition.

Why each choice is right or wrong

A - Correct: Spanning includes every input vertex.

B - Incorrect: Tied weights may yield multiple MSTs.

C - Correct: A tree is a particular kind of graph.

D - Incorrect: An MST is acyclic.

Primary note proof

Study-note section DAA23

> For a connected weighted undirected graph, an MST spans every vertex without cycles and minimizes total edge weight; it has V−1 edges. Negative edges are allowed. Equal weights may yield multiple MSTs; distinct weights suffice for uniqueness. A tree is itself a graph.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Design and Analysis of Algorithms module | PDF page 36 | lines 16-33

> o Minimum Cost Spanning Tree- Prim's Algorithm ✓ Let G(V,E) be an undirected connected graph with vertices v and edge E . ✓ A sub-graph t=(V,E) of the G is a Spanning tree of G iff t is a tree.3 ✓ The problem is to generate a graph G = (V,E) where E is the subset of E,G is a Minimum spanning tree. ✓ Each and every edge will contain the given non-negative length. connect all the nodes with edge present in set E and weight has to be minimum. NOTE: ✓ We have to visit all the nodes. ✓ The subset tree (i.e) any connected graph with N vertices must have at least N-1 edges and also it does not form a cycle. Definition: 1. A spanning tree of a graph is an undirected tree consisting of only those edge that are necessary to connect all the vertices in the original graph. 2. A Spanning tree has a property that for any pair of vertices there exist only one path between them and the insertion of an edge to a spanning tree form a unique cycle.

Design and Analysis of Algorithms module | PDF page 85 | lines 14-20

> 4.7. BASIC SEARCH AND TRAVERSAL TECHNIQUE DEFINING GRAPH: A graphs g consists of a set V of vertices (nodes) and a set E of edges (arcs). We write G=(V,E). V is a finite and non-empty set of vertices. E is a set of pair of vertices; these pairs are called as edges. Therefore, V(G) is read as V of G, is a set of vertices and E(G),read as E of G is a set of edges. An edge e= (v, w) is a pair of vertices v and w, and to be incident with v and w.

Question sources

Computer Science Exit Exam practice collection · Question 116

Question 83 | NS0294

At a fixed input size n, what is the minimum running time over all legal inputs of that size called?

A. Average time

B. Small time

C. Worst time

D. Best time

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer D: Best time

Best-case time is the minimum cost among legal inputs of a fixed size. Worst case takes the maximum, and average case takes an expectation under a specified distribution. Choosing a smaller input size would ask a different question and would not define best case. The revised wording explicitly holds n fixed, making best time the unique standard case term.

Why each choice is right or wrong

A - Incorrect: Average time takes an expectation rather than a minimum.

B - Incorrect: Small time is not the defined input-case category.

C - Incorrect: Worst time takes the maximum at fixed size.

D - Correct: Best time takes the minimum at fixed size.

Primary note proof

Study-note section DAA03

> At fixed size n, best case minimizes cost, worst case maximizes it, and average case takes an expectation under a specified input distribution. Randomized expected cost may average internal choices for one fixed input. Smallest input size is not best case.

Question sources

Computer Science Exit Exam · 2018 · Question 11

Question 84 | NS0314

A researcher solves mergesort's recurrence $T(n)=2T(n/2)+\Theta(n)$, obtaining $\Theta(n \log n)$, and claims that this calculation alone proves every comparison-based sorting algorithm requires $\Omega(n \log n)$ worst-case time. What is the flaw in that reasoning?

A. The proof fails because comparison decision trees allow only array indexing.

B. The recurrence describes one algorithm; a separate argument is needed to lower-bound the whole comparison class.

C. The proof fails only because Θ notation ignores constant factors.

D. There is no flaw: any upper bound for one comparison sort automatically lower-bounds all comparison sorts.

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer B: B. One algorithm's recurrence is not a universal lower-bound argument.

Solving the given recurrence establishes mergesort's $\Theta(n \log n)$ running time, but it does not rule out a different comparison algorithm with less work. The unsupported step is extending one algorithm's recurrence to a lower bound for the whole class. A separate decision-tree argument supplies the universal result: $n!$ possible orders require height at least $\log_2(n!)$. Discarding constants and permitting array indexing do not invalidate that argument.

Why each choice is right or wrong

A - Incorrect: The comparison model does not impose the stated array-indexing-only restriction.

B - Correct: The recurrence bounds mergesort; the $n!$-leaf decision-tree argument is what supplies a class-wide lower bound.

C - Incorrect: Ignoring constants is not the logical error; the unsupported extension from one algorithm to every algorithm is.

D - Incorrect: An upper bound for one available algorithm does not exclude a faster different algorithm, so it is not automatically a universal lower bound.

Primary note proof

Study-note section DAA11

> A comparison decision tree needs at least n! leaves for distinct-key orderings. Height h allows at most 2^h leaves, so h≥log 2(n!)=Ω(n log n); merge sort matches this worst-case bound. Array indexing is not forbidden.

Study-note section DAA14

> Ordinary array merge sort has $T(n)=2T(n/2)+\Theta(n)=\Theta(n \log n)$ across input orders, $\Theta(n)$ auxiliary buffer and $\Theta(\log n)$ recursion depth. A written $+O(n)$ recurrence alone establishes an upper bound; the tight result also uses linear merge work. Empty/singleton subarrays are base cases.

Question sources

Computer Science Exit Exam · 2018 · Question 38

Question 85 | NS0317

A routine sums the integers from n down to 1. For integer n≥0, use mathematical-integer RAM pseudocode with exact unit-cost arithmetic and no overflow. Charge 1 unit for each initialization, comparison, compound decrement i-- and return; charge 1 for addition and 1 for assignment in sum=sum+i. Ignore declarations, parameter passing and function-call overhead. What is T(n)? total(n) { sum=0; for(i=n;i>0;i--) sum=sum+i; return sum; }

A. 5n+5

B. $4n^2+4$

C. 6n+2

D. 4n+4

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer D: 4n+4

The descending loop executes n times. Initializing sum and i costs 2 units; i>0 is tested n+1 times; the compound decrement costs n units; and adding i then assigning the result costs 2n. One return gives T(n)=2+(n+1)+n+2n+1=4n+4. Although the returned sum grows quadratically with n, the number of unit-cost operations grows linearly. Exact mathematical integers prevent accumulator or loop-control overflow.

Why each choice is right or wrong

A - Incorrect: This overcounts the specified model by one unit per iteration and one fixed unit.

B - Incorrect: The numeric sum grows quadratically, but there are only n constant-cost loop iterations.

C - Incorrect: The stated iteration has one test, one compound decrement and two accumulator operations, not six units of iteration work.

D - Correct: Two initializations, n+1 tests, n decrements, 2n body units and one return total 4n+4.

Primary note proof

Study-note section DAA06

> Exact coefficients depend on whether tests, increments, arithmetic and assignments count separately.

Study-note section DAA06

For sum=0; for(i=1;i<=n;i++) sum=sum+1; return sum: counting two initializations, n+1 tests, n increments,2n body operations and one return gives 4n+4. A different primitive-cost convention changes this exact expression, not Θ(n).

Study-note section DAA02

A RAM model commonly gives constant cost to bounded-word arithmetic/access.

Question sources

Computer Science Exit Exam · 2018 · Question 42

Question 86 | NS0468

Which strategy breaks a problem into smaller largely independent subinstances, solves them recursively and combines their results, without caching repeated overlapping states?

A. Greedy

B. Backtracking

C. Divide and Conquer

D. Dynamic Programming

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer C: Divide and Conquer

Divide and conquer solves smaller largely independent instances and combines their results into the original solution. Dynamic programming can also recurse and combine values, but its distinguishing mechanism is reusable state values when subproblems overlap. Greedy is defined by safe irrevocable local choices, and backtracking by revising partial candidates. Independence and the lack of overlap-based caching resolve the original broad decomposition wording.

Why each choice is right or wrong

A - Incorrect: Greedy is defined by safe local choices.

B - Incorrect: Backtracking extends and retracts candidates.

C - Correct: Independent recursive subinstances and combination define divide and conquer.

D - Incorrect: The specified independence excludes the overlapping-state reuse emphasized by DP.

Primary note proof

Study-note section DAA12

> Divide and conquer solves smaller largely independent instances and combines results. DP reuses a defined set of subproblem states. Greedy commits to a feasible local choice only when a safety argument supports it. Backtracking grows/retracts candidates when constraints fail; branch and bound also rejects candidates whose best possible objective cannot beat an incumbent.

Question sources

Algorithm Design & Analysis review questions · Question 1

Question 87 | NS0469

What is the formal definition of Big-O notation? State the roles of constants c and k.

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: $f \in O(g)$ iff $\exists c>0, k$ such that $0 \le f(n) \le c\, g(n)$ for all $n \ge k$.

DAA04 requires one fixed positive multiplier and one threshold covering the entire tail of the eventually nonnegative functions. c permits constant-factor scaling; k excludes a finite initial prefix. Neither may be chosen anew for each n. This defines an eventual upper bound, independently of input case or whether the resource is time or space.

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, $f \in O(g)$ means $f(n) \le c \cdot g(n)$ for all $n \ge n_0$, with fixed $c>0$. Ω reverses that inequality; Θ requires both bounds. Little-o means $f/g \to 0$, and little-ω means $f/g \to \infty$ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Question sources

Algorithm Design & Analysis review questions · Question 2

Question 88 | NS0470

Show that f(n) = 10n + 5 is O(n). What values of c and k satisfy the definition?

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: c=15 and k=1 work.

For every $n \geq 1$, $5 \leq 5n$, so $10n+5 \leq 15n$. Choosing $c=15$ and $k=1$ therefore satisfies the Big-O upper inequality for the whole tail. DAA04 gives this exact worked example. These witnesses need not be unique; the goal is one valid fixed constant and threshold, not the smallest pair.

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, $f \in O(g)$ means $f(n) \leq c \cdot g(n)$ for all $n \geq n_0$, with fixed $c>0$. Ω reverses that inequality; Θ requires both bounds. Little-o means $f/g \to 0$, and little-ω means $f/g \to \infty$ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Study-note section DAA04

> $10n+5 \leq 15n$ for $n \geq 1$, so $c=15, n_0=1$ works.

Question sources

Algorithm Design & Analysis review questions · Question 3

Question 89 | NS0471

Show that $f(n) = 3n^2 + 4n + 1$ is $O(n^2)$. What are the constants?

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: c=8 and k=1 work.

For $n \geq 1$, replace $4n$ by at most $4n^2$ and 1 by at most n^2. The sum is then at most $8n^2$, satisfying DAA04's upper-bound definition. Thus $c=8$ and $k=1$ are valid witnesses. They are not uniquely required constants; larger multipliers or thresholds can also satisfy the inequality.

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, $f \in O(g)$ means $f(n) \leq c \cdot g(n)$ for all $n \geq n_0$, with fixed $c>0$. Ω reverses that inequality; Θ requires both bounds. Little-o means $f/g \to 0$, and little-ω means $f/g \to \infty$ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Study-note section DAA04

> Also $3n^2+4n+1 \leq 8n^2$ for $n \geq 1$.

Question sources

Algorithm Design & Analysis review questions · Question 4

Question 90 | NS0472

Rank these complexity classes from slowest to fastest growing: O(n!), O(n^2), O(log n), O(2^n), O(n log n)

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: Increasing representative growth: log n, n log n, n^2, 2^n, n!.

DAA05 orders logarithmic, polynomial, exponential, and factorial rates. A log factor above n is still dominated by n^2; fixed-base exponential eventually exceeds any fixed polynomial, and factorial exceeds fixed-base exponential. This ranks representative functions, not mutually exclusive Big-O sets, since a slower function can also satisfy a looser larger upper bound.

Primary note proof

Study-note section DAA05

> Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Question sources

Algorithm Design & Analysis review questions · Question 5

Question 91 | NS0473

Big-Theta Θ(h(n)) provides what kind of bound on an algorithm's complexity?

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: A tight asymptotic bound: both upper and lower constant-factor bounds.

DAA04 requires positive constants c1,c2 and a threshold k with c1 h(n)≤f(n)≤c2 h(n) for all n≥k. Theta therefore specifies growth up to constant factors, not exact seconds. It can describe best, average, worst, or space functions; choosing Theta does not itself choose an input case.

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, f∈O(g) means f(n)≤c·g(n) for all n≥n 0, with fixed c>0. Ω reverses that inequality; Θ requires both bounds. Little-o means f/g→0, and little-ω means f/g→∞ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Question sources

Algorithm Design & Analysis review questions · Question 6

Question 92 | NS0474

What does Big-Omega Ω(g(n)) represent and how does it differ from Big-O?

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: Ω is an eventual lower bound; O is an eventual upper bound.

DAA04 gives $f \in \Omega(g)$ when $f(n) \geq c\, g(n)$ beyond a fixed threshold for some $c>0$, reversing the Big-O inequality. The relations describe functions independently of best or worst case. A function may satisfy both, giving Theta, or only one for a particular comparator; they are not literal fastest-versus-slowest input labels.

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, $f \in O(g)$ means $f(n) \leq c \cdot g(n)$ for all $n \geq n_0$, with fixed $c>0$. Ω reverses that inequality; Θ requires both bounds. Little-o means $f/g \to 0$, and little-ω means $f/g \to \infty$ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Question sources

Algorithm Design & Analysis review questions · Question 7

Question 93 | NS0475

State the Transitivity Theorem of Big-O and explain why it is useful.

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: If f∈O(g) and g∈O(h), then f∈O(h).

DAA04 permits chained upper bounds. If f≤c g past k1 and g≤d h past k2, then f≤cd h for every n beyond max(k1,k2). Multiplying constants and combining thresholds proves transitivity. It is useful for replacing a complicated intermediate expression by a simpler known bound without redoing the entire comparison.

Primary note proof

Study-note section DAA04

> Prove the inequality for the whole tail, not selected values. Transitivity allows chained upper bounds; sums are dominated by the largest nonnegative growth term.

Study-note section DAA04

> For eventually nonnegative f, g, f∈O(g) means f(n)≤c·g(n) for all n≥n 0, with fixed c>0. Ω reverses that inequality; Θ requires both bounds. Little-o means f/g→0, and little-ω means f/g→∞ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Question sources

Algorithm Design & Analysis review questions · Question 8

Question 94 | NS0476

According to Big-O Theorem 6, order the following correctly: n!, 2^n, n^2, n log n, n, log n, constant k

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: Increasing growth: k, log n, n, n log n, n^2, 2^n, n!.

For fixed positive k, DAA05 places constant growth first, followed by logarithmic and linear rates, then n log n, quadratic, exponential, and factorial growth. The ordering is eventual, not a promise of speed for every small input. It concerns representative functions; Big-O memberships can overlap when upper bounds are loose.

Primary note proof

Study-note section DAA05

> Representative growth is 1,log n,n,n log n,n^2,n^3,...,b^n,n! for fixed b>1. Log bases differ by a constant. Both n log n and n^2 are superlinear; log n is sublinear. Polynomial growth has a fixed exponent in input size.

Question sources

Algorithm Design & Analysis review questions · Question 9

Question 95 | NS0477

How is algorithm validation different from algorithm analysis?

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: Validation checks required behavior; analysis measures resource costs.

DAA01 specifies legal inputs, required outputs, and correctness proof through invariants or recursive induction. DAA02 counts operations and storage as input size grows. A procedure can be fast yet incorrect, or correct yet expensive. Testing can reveal failures but finite tests do not prove correctness for all inputs, so resource analysis does not replace validation.

Primary note proof

Study-note section DAA01

> State inputs, output, constraints and empty-solution policy. Algorithms are finite, definite executable methods independent of source language; feasible satisfies constraints, optimal is best feasible. Conditional branches are legitimate control flow.

Study-note section DAA02

> Analysis counts operations and storage as functions of input size: array length n, graph vertices V/edges E, or the number of bits in integers. A RAM model commonly gives constant cost to bounded-word arithmetic/access. Arbitrarily large integer arithmetic and printing long outputs need their own costs. Elapsed measurements also depend on hardware, compiler, cache, I/O and competing load.

Question sources

Algorithm Design & Analysis review questions · Question 10

Question 96 | NS0478

What is the general recurrence equation for a divide-and-conquer algorithm?

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: For equal-size splits, T(n)=aT(n/b)+f(n), with an explicit base case.

DAA10 uses a recursive subcalls of size n/b and f(n) nonrecursive divide/combine work, with a≥1 and b>1. Include a base such as T(1)=Θ(1). Unequal or decrementing subproblems require a more general sum of subproblem costs; the equal-split form is a common model, not every possible divide-and-conquer recurrence.

Primary note proof

Study-note section DAA10

> For T(n)=aT(n/b)+f(n), a≥1, b>1, let p=log_b a. Level i has a^i nodes of size n/b^i. Compare f with n^p: a polynomial gap below gives Θ(n^p); f=Θ(n^p) gives Θ(n^p log n); a polynomial gap above gives Θ(f(n)) if a f(n/b)≤c f(n) for fixed c<1. The third case needs this regularity condition.

Question sources

Algorithm Design & Analysis review questions · Question 11

Question 97 | NS0479

What is the time complexity of Merge Sort and how is it derived?

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: Ordinary array merge sort is Θ(n log n).

DAA14 gives $T(n)=2T(n/2)+\Theta(n)$: two half-size sorts and linear merging. There are logarithmically many levels, each totaling linear merge work, with linear total leaf work. Therefore the tight result is n log n across input orders, assuming the ordinary algorithm rather than a variant that skips already ordered merges.

Primary note proof

Study-note section DAA14

> Ordinary array merge sort has $T(n)=2T(n/2)+\Theta(n)=\Theta(n \log n)$ across input orders, $\Theta(n)$ auxiliary buffer and $\Theta(\log n)$ recursion depth. A written $+O(n)$ recurrence alone establishes an upper bound; the tight result also uses linear merge work. Empty/singleton subarrays are base cases.

Question sources

Algorithm Design & Analysis review questions · Question 12

Question 98 | NS0480

Explain the three steps of Merge Sort in order.

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: Split into halves, recursively sort each half, then merge the sorted halves.

DAA14 stops at empty or singleton ranges. During combination it repeatedly takes the smaller remaining front item, then appends the unexhausted suffix. Left-first equal-key selection preserves stability. These steps separate division, recursive solution, and combination; the merge assumes each half is already sorted rather than that the original full array was sorted.

Primary note proof

Study-note section DAA14

> Split by indices, recursively sort each half, then repeatedly copy the smaller remaining front element into a temporary result. The merge invariant says the output prefix is sorted and contains the smallest elements already consumed. Append the unexhausted suffix. Equal keys must be taken from the left half first to preserve stability.

Question sources

Algorithm Design & Analysis review questions · Question 13

Question 99 | NS0481

What property makes Merge Sort a 'stable' sorting algorithm?

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: Take equal keys from the left half first while merging.

DAA14 explicitly requires left-first equal-key handling for stability. Recursive halves retain their internal equal-key order, and choosing a left equal before a right equal preserves their original cross-half order. Inductively the entire output retains the relative order of equal-key records. Choosing the right equal first can break that property.

Primary note proof

Study-note section DAA14

> Split by indices, recursively sort each half, then repeatedly copy the smaller remaining front element into a temporary result. The merge invariant says the output prefix is sorted and contains the smallest elements already consumed. Append the unexhausted suffix. Equal keys must be taken from the left half first to preserve stability.

Question sources

Algorithm Design & Analysis review questions · Question 14

Question 100 | NS0482

What is the average and worst-case time complexity of Quick Sort, and when does the worst case occur?

Hint

Match the defining mechanism and its assumptions to the required claim; keep resource bounds separate from correctness.

Answer: Expected average $\Theta(n \log n)$; worst $\Theta(n^2)$ from repeated highly unbalanced partitions.

DAA15 gives logarithmic partition depth when splits are balanced and expected n log n with random pivots. Repeated extreme pivots leave sizes $n-1, n-2, \ldots$, causing $n+(n-1)+\ldots$ work. Sorted input can trigger this with certain fixed endpoint pivots, but input order alone is not a universal worst-case characterization across partition rules.

Primary note proof

Study-note section DAA15

> Balanced splits give $\Theta(n \log n)$; repeatedly highly unbalanced splits give $\Theta(n^2)$. Random pivots give expected $\Theta(n \log n)$, not a deterministic guarantee. Median-of-three does not eliminate adversarial cases; three-way partitioning helps repeated keys. Standard quicksort is unstable, uses expected $\Theta(\log n)$ stack but worst $\Theta(n)$; recursing on the smaller side and iterating on the larger bounds stack by $O(\log n)$.

Question sources

Algorithm Design & Analysis review questions · Question 15

Question 101 | NS0483

Describe the partitioning step of Quick Sort using indices i and j.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Advance i over values below the pivot and j over values above; swap while i≤j, then advance both.

DAA15's two-pointer partition bounds its scans and ensures progress even on equal values. After pointers cross, recurse on the resulting smaller ranges. Hoare-style boundaries need not put the pivot in its final position, whereas Lomuto commonly does. The chosen return convention and recursion ranges must match; mixing them can miss elements or fail to terminate.

Primary note proof

Study-note section DAA15

> Choose a pivot, partition, then sort the resulting smaller ranges. In a two-pointer scheme, advance i while values are below pivot and j while above; when i≤j, swap and advance both. Bound the scans and ensure progress on equal values. Recurse on the resulting ranges. Hoare-style partition boundaries need not put the pivot at its final position; Lomuto-style partition commonly does. Do not mix their return conventions.

Question sources

Algorithm Design & Analysis review questions · Question 16

Question 102 | NS0484

Compare Merge Sort and Quick Sort on: stability, worst-case complexity, and space usage.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Merge: stable, Θ(n log n) worst, Θ(n) buffer. Quick: unstable, Θ(n^2) worst, expected Θ(log n) and worst Θ(n) stack.

DAA14 provides merge's linear buffer and logarithmic recursion depth, with left-first ties for stability. DAA15 gives quicksort's in-array partitioning but nonconstant recursive storage and potentially linear depth. Recursing only on the smaller side can bound that stack logarithmically. These describe ordinary implementations; ignoring recursive frames would misstate total auxiliary space.

Primary note proof

Study-note section DAA14

> Ordinary array merge sort has T(n)=2T(n/2)+Θ(n)=Θ(n log n) across input orders, Θ(n) auxiliary buffer and Θ(log n) recursion depth. A written +O(n) recurrence alone establishes an upper bound; the tight result also uses linear merge work. Empty/singleton subarrays are base cases.

Study-note section DAA15

> Balanced splits give Θ(n log n); repeatedly highly unbalanced splits give Θ(n^2). Random pivots give expected Θ(n log n), not a deterministic guarantee. Median-of-three does not eliminate adversarial cases; three-way partitioning helps repeated keys. Standard quicksort is unstable, uses expected Θ(log n) stack but worst Θ(n); recursing on the smaller side and iterating on the larger bounds stack by O(log n).

Study-note section DAA14

> Equal keys must be taken from the left half first to preserve stability.

Question sources

Algorithm Design & Analysis review questions · Question 17

Question 103 | NS0485

Why is Binary Search classified as a Divide and Conquer algorithm?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Each midpoint comparison reduces the ordered-search instance to one half.

DAA13 maintains sorted low/high bounds and excludes the half that cannot contain the target. The recurrence $T(n)=T(n/2)+\Theta(1)$ gives logarithmic worst work. Unlike sorting two halves, binary search solves only one surviving subinstance and requires no substantial combination step; this is the common divide-and-conquer or decrease-by-factor classification.

Primary note proof

Study-note section DAA13

> Binary search requires sorted random-access data. Maintain inclusive low/high bounds, compute mid=low+(high−low)//2, and replace high with mid−1 or low with mid+1 according to the comparison. Stop when found or low>high. $T(n)=T(n/2)+\Theta(1)$: worst $\Theta(\log n)$, best $\Theta(1)$; iterative auxiliary space $\Theta(1)$, recursive $\Theta(\log n)$.

Study-note section DAA12

> Divide and conquer solves smaller largely independent instances and combines results.

Question sources

Algorithm Design & Analysis review questions · Question 18

Question 104 | NS0486

What is the core principle of the Greedy algorithmic paradigm?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Repeatedly make a locally attractive feasible choice and keep it.

DAA19 defines greedy decisions as irrevocable rather than exploring and undoing alternatives. To obtain a global optimum, the local choice needs a problem-specific safety argument, often an exchange proof. Optimal substructure alone is insufficient. Thus local attractiveness is the mechanism, while global optimality is a separate correctness obligation rather than an automatic consequence.

Primary note proof

Study-note section DAA19

> A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it. Optimal substructure alone does not prove this safe. An exchange proof inserts the greedy choice into an optimum without worsening cost; recurse. MST cut proofs use this exchange.

Question sources

Algorithm Design & Analysis review questions · Question 19

Question 105 | NS0487

What is the difference between a feasible solution and an optimal solution in the context of Greedy algorithms?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Feasible satisfies constraints; optimal is best among feasible solutions.

DAA01 separates constraint satisfaction from objective quality. For example, a legal set of knapsack items may fit capacity but have less value than another legal set. DAA19 warns that greedy choices require a safety proof to reach an optimum; constructing a feasible result alone does not prove global optimality.

Primary note proof

Study-note section DAA01

> State inputs, output, constraints and empty-solution policy. Algorithms are finite, definite executable methods independent of source language; feasible satisfies constraints, optimal is best feasible. Conditional branches are legitimate control flow.

Study-note section DAA19

> A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it. Optimal substructure alone does not prove this safe. An exchange proof inserts the greedy choice into an optimum without worsening cost; recurse. MST cut proofs use this exchange.

Question sources

Algorithm Design & Analysis review questions · Question 20

Question 106 | NS0488

Name two advantages and two disadvantages of Greedy algorithms.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Advantages: simple choices and low overhead. Disadvantages: irrevocable mistakes and problem-specific applicability.

DAA19 identifies these benefits and limitations. Greedy can avoid maintaining a large state table, but a locally attractive choice can prevent an optimum and cannot later be undone within the paradigm. An exchange or other safety proof is therefore essential. A greedy heuristic has no approximation guarantee solely because it uses local choices.

Primary note proof

Study-note section DAA19

> Advantages are often simple decisions and low overhead; limitations are irrevocable errors and problem-specific applicability. A heuristic has no approximation guarantee merely because it is greedy. For coins{1,3,4} and target 6, largest-first gives 4+1+1 while 3+3 uses fewer coins. Activity selection instead has a safe earliest-finish rule for maximizing the number of nonoverlapping unweighted intervals; replacing the first chosen activity in an optimum with an earlier-finishing one leaves at least as much room.

Question sources

Algorithm Design & Analysis review questions · Question 21

Question 107 | NS0489

In the Job Sequencing with Deadlines algorithm, why are jobs sorted in decreasing order of profit first?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Profit-first gives high-value feasible jobs priority under the unit-duration scheduling assumptions.

DAA21 processes nonnegative-profit jobs in decreasing profit and places each in its latest free slot meeting its deadline. This preserves earlier slots for tighter deadlines, while an exchange argument supports profit-first selection for feasible unit-job sets. The rule needs one machine, initially available unit jobs, and on-time profit; variable durations can invalidate it.

Primary note proof

Study-note section DAA21

> Assume one machine, initially available jobs, unit durations, integer deadlines and profit only for on-time completion. Process nonnegative-profit jobs by decreasing profit; place each in the latest free slot finishing by its deadline, or reject it. Latest placement preserves earlier slots for tighter deadlines; feasible unit-job sets support profit-first exchange.

Question sources

Algorithm Design & Analysis review questions · Question 22

Question 108 | NS0490

Given jobs J4(deadline=2, profit=300), J1(deadline=5, profit=200), J3(deadline=3, profit=190), J2(deadline=3, profit=180), J5(deadline=4, profit=120), J6(deadline=2, profit=100) - what is the optimal schedule and maximum profit?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Slots 1-5: J2,J4,J3,J5,J1; maximum profit 990.

DAA21's greedy placement gives J4 at slot2, J1 at5, J3 at3, J2 at1, and J5 at4; J6 cannot fit before deadline2. Every accepted deadline holds. Their profits sum to 180+300+190+120+200=990. Since these are the five largest profits and only five slots exist, 990 also matches an upper bound.

Primary note proof

Study-note section DAA21

> Assume one machine, initially available jobs, unit durations, integer deadlines and profit only for on-time completion. Process nonnegative-profit jobs by decreasing profit; place each in the latest free slot finishing by its deadline, or reject it. Latest placement preserves earlier slots for tighter deadlines; feasible unit-job sets support profit-first exchange.

Study-note section DAA21

> Slots finishing 1…5 contain J 2,J 4,J 3,J 5,J 1. Profits 180+300+190+120+200=990; deadlines hold. J 6 cannot finish by 2. The five largest profits sum to 990, giving a matching upper bound.

Question sources

Algorithm Design & Analysis review questions · Question 23

Question 109 | NS0491

What is the Optimal Merge Pattern and what greedy strategy does it apply?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Repeatedly merge the two smallest files or weights, reinsert their sum, and minimize total merge cost.

DAA22 charges p+q for combining file sizes p and q, adds that charge to total cost, and returns the merged size to a min-heap. Combining small weights early avoids repeatedly charging large weights at deep levels. The min-heap implementation costs O(n log n); actual comparison counts are a distinct cost model.

Primary note proof

Study-note section DAA22

> Combine the two smallest weights, add their sum to total cost, and reinsert it. A min-heap gives O(n log n) time/O(n) space. Cost p+q counts moved records; worst-case comparisons are p+q−1 for nonempty lists, and actual comparisons may be fewer.

Question sources

Algorithm Design & Analysis review questions · Question 24

Question 110 | NS0492

What is a Minimum Spanning Tree (MST) and what problem does it solve?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: An MST connects all vertices without cycles using minimum total edge weight.

DAA23 assumes a connected undirected weighted graph and gives an MST V−1 edges. It minimizes total connection cost, not every source-to-vertex route. Prim and Kruskal find such trees through safe light edges. Equal weights can allow several optima, and disconnected input instead yields a minimum spanning forest rather than one spanning tree.

Primary note proof

Study-note section DAA23

> For a connected weighted undirected graph, an MST spans every vertex without cycles and minimizes total edge weight; it has V−1 edges. Negative edges are allowed. Equal weights may yield multiple MSTs; distinct weights suffice for uniqueness. A tree is itself a graph.

Question sources

Algorithm Design & Analysis review questions · Question 25

Question 111 | NS0493

Compare Prim's Algorithm and Kruskal's Algorithm: approach, edge selection, and best use case.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Prim grows one tree across a cut; Kruskal accepts globally light edges joining distinct components.

DAA23 gives matrix Prim $O(V^2)$, heap/list Prim $O(E \log V)$, and edge-sorting Kruskal $O(E \log E)$. Matrix Prim can favor dense graphs because it avoids sorting quadratically many edges; Kruskal naturally uses an edge list and union-find. Best use depends on representation and implementation, not a universal measured-speed ranking.

Primary note proof

Study-note section DAA23

> Prim repeatedly adds the cheapest edge crossing from its current tree to an outside vertex. Kruskal sorts edges globally, accepting one only when union-find says its endpoints are in different components. The cut proof exchanges a light crossing edge into some optimum without raising its weight. Matrix Prim costs $O(V^2)$; binary-heap Prim $O(E \log V)$, and Kruskal $O(E \log E)$. Dense-graph comparisons depend on these implementations. Disconnected input gives a minimum spanning forest.

Question sources

Algorithm Design & Analysis review questions · Question 26

Question 112 | NS0494

How does Kruskal's Algorithm detect and avoid cycles when building the MST?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Use union-find: reject an edge if its endpoints already belong to the same component.

DAA23 processes edges globally in increasing weight. If the endpoint representatives differ, the edge joins components without creating a cycle, so accept it and union the sets. If they match, a path already connects the endpoints and adding the edge would close a cycle. The light-edge exchange property supports minimum total weight.

Primary note proof

Study-note section DAA23

> Prim repeatedly adds the cheapest edge crossing from its current tree to an outside vertex. Kruskal sorts edges globally, accepting one only when union-find says its endpoints are in different components. The cut proof exchanges a light crossing edge into some optimum without raising its weight. Matrix Prim costs $O(V^2)$; binary-heap Prim O(E log V), and Kruskal O(E log E). Dense-graph comparisons depend on these implementations. Disconnected input gives a minimum spanning forest.

Question sources

Algorithm Design & Analysis review questions · Question 27

Question 113 | NS0495

What is Dijkstra's Algorithm used for and what is its key limitation?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Single-source shortest paths with nonnegative edge weights; negative edges break its general guarantee.

DAA24 finalizes the least unsettled tentative distance and relaxes outgoing edges. Nonnegative weights ensure a later route through unsettled vertices cannot improve a finalized distance. A negative edge can violate that proof even without a negative cycle. This differs from MST construction, and unreachable vertices retain infinite distance.

Primary note proof

Study-note section DAA24

> Nonnegative edges ensure a later path through unsettled vertices cannot improve a finalized distance. With $s \to a=2$, $s \to b=5$, $b \to a=-4$, finalizing a at 2 misses the later distance 1. Binary-heap/list implementation costs $O((V+E)\log V)$, matrix scanning $O(V^2)$. An MST minimizes total connection cost; Dijkstra minimizes each source-path distance, a different objective.

Question sources

Algorithm Design & Analysis review questions · Question 28

Question 114 | NS0496

List the steps of Dijkstra's Algorithm in order.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Initialize distances; repeatedly finalize the smallest unsettled distance and relax its outgoing edges.

DAA24 starts source at zero, all others at infinity, with predecessors unset. Extract the minimum unsettled vertex; update dist[v] and its predecessor when dist[u]+w(u,v) improves it. Stop when none remain or the minimum is infinite. For an early destination stop, finalize it first rather than stopping merely on discovery.

Primary note proof

Study-note section DAA24

> Initialize dist[source]=0, others∞ and predecessors unset. Repeatedly extract the unsettled vertex u of minimum distance; relax each edge by dist[v]=min(dist[v], dist[u]+w(u, v)), updating predecessors when improved. Once the minimum is∞, remaining vertices are unreachable. A destination may be finalized before stopping, not merely discovered.

Question sources

Algorithm Design & Analysis review questions · Question 29

Question 115 | NS0497

What is the difference between MST (Minimum Spanning Tree) and SSSP (Single-Source Shortest Path)?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: MST minimizes total connection weight; SSSP minimizes each source-to-vertex path distance.

DAA23's spanning tree connects every vertex with minimum sum of selected edge weights. DAA24's source shortest paths optimize a different objective for a designated source and can choose a different tree. An MST need not contain shortest source routes. Standard MST permits negative weights, while Dijkstra's particular SSSP algorithm needs nonnegative weights.

Primary note proof

Study-note section DAA23

> For a connected weighted undirected graph, an MST spans every vertex without cycles and minimizes total edge weight; it has V−1 edges. Negative edges are allowed. Equal weights may yield multiple MSTs; distinct weights suffice for uniqueness. A tree is itself a graph.

Study-note section DAA24

> Nonnegative edges ensure a later path through unsettled vertices cannot improve a finalized distance. With s→a=2, s→b=5, b→a=−4, finalizing a at 2 misses the later distance 1. Binary-heap/list implementation costs $O((V+E)\log V)$, matrix scanning $O(V^2)$. An MST minimizes total connection cost; Dijkstra minimizes each source-path distance, a different objective.

Question sources

Algorithm Design & Analysis review questions · Question 30

Question 116 | NS0498

Which two textbook structural properties justify and motivate the usual dynamic-programming approach to optimization? Explain their different roles rather than treating them as universal requirements for every use of DP.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Optimal substructure and overlapping subproblems: the first justifies composing optimal subsolutions, while the second motivates reusing stored state results.

Optimal substructure means an optimal solution can be built using appropriate optimal subsolutions. Overlap means the same state problem occurs repeatedly, making saved results useful. These properties justify and motivate the familiar optimization pattern, provided the state captures all relevant information and the recurrence is correct. DP can also count or decide, so the two properties are not stated here as universal necessities for every DP computation.

Primary note proof

Study-note section DAA25

> Overlapping states motivate reuse; optimal substructure supports optimization. These are textbook signatures, not requirements that every DP problem optimize or every state have multiple users. DP also counts/decides. Cost=states×work per state and can remain exponential.

Question sources

Algorithm Design & Analysis review questions · Question 31

Question 117 | NS0499

How does Dynamic Programming differ from Divide and Conquer?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Divide and conquer solves largely independent instances; DP reuses dependent or overlapping states.

DAA12 distinguishes independent recursive decomposition and combination from a defined set of reused states. DP requires state meaning, transitions, bases, and dependency order so each needed state is computed without redundant recomputation. Divide and conquer usually does not need a shared-state cache. Merely splitting a problem into smaller parts is common to both and does not distinguish them.

Primary note proof

Study-note section DAA12

> Divide and conquer solves smaller largely independent instances and combines results. DP reuses a defined set of subproblem states. Greedy commits to a feasible local choice only when a safety argument supports it. Backtracking grows/retracts candidates when constraints fail; branch and bound also rejects candidates whose best possible objective cannot beat an incumbent.

Study-note section DAA25

> Define state meaning, choices, recurrence, bases and dependency order. Prove replacing a subsolution by a better one improves the full solution; otherwise state information may be missing. Show the recurrence covers all legal final choices and generates feasible candidates, then induct in dependency order. Store choices for reconstruction.

Question sources

Algorithm Design & Analysis review questions · Question 32

Question 118 | NS0500

How does Dynamic Programming differ from the Greedy approach?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Greedy commits to a proved-safe local choice; DP compares stored subproblem alternatives.

DAA19 makes greedy choices irrevocable and requires a safety argument. DAA25 defines states and recurrence choices, then computes optimal state values in dependency order. DP can retain alternatives that a greedy rule would discard. Optimal substructure does not itself make greedy safe, and neither approach is universally preferable without the problem's structural conditions.

Primary note proof

Study-note section DAA19

> A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it. Optimal substructure alone does not prove this safe. An exchange proof inserts the greedy choice into an optimum without worsening cost; recurse. MST cut proofs use this exchange.

Study-note section DAA25

> Define state meaning, choices, recurrence, bases and dependency order. Prove replacing a subsolution by a better one improves the full solution; otherwise state information may be missing. Show the recurrence covers all legal final choices and generates feasible candidates, then induct in dependency order. Store choices for reconstruction.

Question sources

Algorithm Design & Analysis review questions · Question 33

Question 119 | NS0501

What are the four development steps of a Dynamic Programming algorithm?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Define and justify optimal states; derive recurrence and bases; compute dependencies once; reconstruct retained choices.

DAA25 first requires a state meaning and legal choices, then a correctness argument for the recurrence and feasible candidates. Evaluate states in dependency order, accounting for states times work per state. Store decisions so the chosen solution can be reconstructed afterward. These provide the requested four development stages without assuming DP is automatically polynomial.

Primary note proof

Study-note section DAA25

> Define state meaning, choices, recurrence, bases and dependency order. Prove replacing a subsolution by a better one improves the full solution; otherwise state information may be missing. Show the recurrence covers all legal final choices and generates feasible candidates, then induct in dependency order. Store choices for reconstruction.

Question sources

Algorithm Design & Analysis review questions · Question 34

Question 120 | NS0502

Explain memoization and tabulation - the two DP implementation strategies.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Memoization solves requested states recursively with a cache; tabulation fills states in explicit dependency order.

DAA26 checks the cache before recursively solving a memoized state and stores its result. Tabulation schedules prerequisite states first without that demand-driven recursion. Keys must include all relevant parameters, and chosen domains may avoid irrelevant states. Both reuse results, but sparsity, recursion overhead, and locality can affect which implementation is preferable.

Primary note proof

Study-note section DAA26

> Memoization recursively checks a cache before solving a state and stores its result. Tabulation orders dependencies explicitly; chosen domains may exclude irrelevant states. Cache keys include all relevant parameters; recursion/locality/sparsity affect implementation cost.

Question sources

Algorithm Design & Analysis review questions · Question 35

Question 121 | NS0503

In the Fibonacci sequence, why is naive recursion inefficient and how does DP improve it?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Naive recursion repeats Fibonacci states; DP computes each required state once.

DAA26 shows that F(n−2) reappears across recursion branches, causing Θ(φ^n) calls, with O(2^n) a valid loose bound. Memoization or tabulation uses the two preceding values and only Θ(n) unit-cost additions. Two rolling values reduce working machine words to constant, though arbitrary-precision values require growing bit costs.

Primary note proof

Study-note section DAA26

> Naive Fibonacci recomputes F(n−2) across branches: call count is Θ(φ^n), with O(2^n) a valid loose upper bound. F(0)=0, F(1)=1 and F(i)=F(i−1)+F(i−2) yield Θ(n) unit-cost additions with memoization/tabulation. Keeping only two recent values uses Θ(1) machine words; arbitrary-precision output needs growing bit storage. This naive overlapping recursion is not the independent-subproblem ideal of D&C.

Question sources

Algorithm Design & Analysis review questions · Question 36

Question 122 | NS0504

What is the 0/1 knapsack optimization problem with positive integer weights, nonnegative values and integer capacity W, and why does its item-and-capacity state formulation suit dynamic programming? Each item may be chosen at most once.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Choose a subset with total weight at most W and maximum total value. DP compares excluding each item with including it once and reuses item/capacity state results.

Define K[i,w] as the best value using the first i items within capacity w. An item that fits can be excluded or included with its value added to K[i-1,w-weight_i]; taking the larger value gives the recurrence. The previous-row reference enforces at-most-once use, and repeated states are reused. The O(nW) running time is pseudo-polynomial in binary-encoded capacity, unlike the separate fractional-knapsack greedy method.

Primary note proof

Study-note section DAA29

> For integer capacity W, K[i, w] is maximum value using the first i items within capacity w. Initialize K[0, w]=0. If weight_i>w, use K[i−1, w]; otherwise take max(K[i−1, w], value_i+K[i−1, w−weight_i]). The previous-row reference ensures each item is used at most once. Backtrack changed values to recover chosen items.

Study-note section DAA20

> This proof depends on divisibility. With capacity 10 and items(6,12),(5,8),(5,8), ratio-greedy chooses value 12; the two weight 5 items give 16 under 0/1 rules. Specify whether the knapsack problem is fractional,0/1 or unbounded. Feasibility alone does not guarantee optimality.

Question sources

Algorithm Design & Analysis review questions · Question 37

Question 123 | NS0505

What is the All Pairs Shortest Path (APSP) problem and which algorithm solves it?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: APSP finds shortest distances for every ordered vertex pair; Floyd-Warshall is one DP algorithm.

DAA28 initializes direct-edge costs and progressively allows each vertex as an intermediate. Its min-plus recurrence gives $\Theta(V^3)$ time and $\Theta(V^2)$ storage. Negative cycles make affected pair distances lack a finite minimum; Floyd detects them through negative diagonals. Its cost is a taught solution rather than a universal best-achievable APSP bound.

Primary note proof

Study-note section DAA28

> For all-pairs distances initialize D[i, i]=0, D[i, j] to the minimum direct edge weight, and missing edges to∞. Preserve a negative self-loop by taking min(0, its weight). For k outermost, update D[i, j]=min(D[i, j], D[i, k]+D[k, j]); before phase k, intermediates are restricted to earlier vertices. Guard unreachable sums and numeric overflow.

Study-note section DAA28

> Three loops give $\Theta(V^3)$ time and $\Theta(V^2)$ storage. A negative diagonal afterward identifies a negative cycle; pairs able to traverse it have no finite minimum walk distance. Warshall reachability replaces min/+ by OR/AND. Floyd's cost is not a universal best-achievable bound for all APSP settings; sparse nonnegative graphs permit repeated Dijkstra and other models permit other methods.

Question sources

Algorithm Design & Analysis review questions · Question 38

Question 124 | NS0506

Describe the APSP initialization step. What values fill the matrix initially?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Initialize D[i,i]=0, direct pairs to minimum edge weights, and missing edges to infinity.

DAA28 uses zero for the empty self-path, but preserves a negative self-loop by taking min(0, its weight). Parallel direct edges use their minimum. Infinity denotes no known route, so later arithmetic must guard unreachable sums and overflow. This is Floyd's initialization before intermediate vertices are considered, not an adjacency-only final answer.

Primary note proof

Study-note section DAA28

> For all-pairs distances initialize D[i, i]=0, D[i, j] to the minimum direct edge weight, and missing edges to∞. Preserve a negative self-loop by taking min(0, its weight). For k outermost, update D[i, j]=min(D[i, j], D[i, k]+D[k, j]); before phase k, intermediates are restricted to earlier vertices. Guard unreachable sums and numeric overflow.

Question sources

Algorithm Design & Analysis review questions · Question 39

Question 125 | NS0507

What is the DP update rule used in the APSP algorithm for each intermediate vertex k?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: D[i,j] ← min(D[i,j], D[i,k]+D[k,j]), with k outermost.

DAA28's phase invariant permits only earlier vertices as intermediates before phase k. Each pair then compares its current route with a route through k. Guard infinite operands and overflow rather than adding unreachable values blindly. Keeping k as the outer phase index is essential to the stated dependency proof.

Primary note proof

Study-note section DAA28

> For all-pairs distances initialize D[i, i]=0, D[i, j] to the minimum direct edge weight, and missing edges to∞. Preserve a negative self-loop by taking min(0, its weight). For k outermost, update D[i, j]=min(D[i, j], D[i, k]+D[k, j]); before phase k, intermediates are restricted to earlier vertices. Guard unreachable sums and numeric overflow.

Question sources

Algorithm Design & Analysis review questions · Question 40

Question 126 | NS0508

Which design strategy explores ALL possible solutions by incrementally building candidates and abandoning invalid ones?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Backtracking

DAA36 assigns the next choice, checks whether a feasible extension remains possible, recurses, and undoes the choice on return. A complete search reports every required candidate while pruning only impossible branches. Stop after one solution only if the task asks for existence or a single solution; otherwise early stopping would fail the ALL requirement.

Primary note proof

Study-note section DAA36

> Represent a partial assignment x[1.. k]. For each allowed next choice: assign it, test whether an extension remains possible, recurse if promising, and undo the assignment on return. Report complete feasible candidates; stop after one only if the task requests existence/a single answer. Explicit constraints restrict each variable’s domain; implicit constraints relate choices.

Question sources

Algorithm Design & Analysis review questions · Question 41

Question 127 | NS0509

Explain why Greedy algorithms are not guaranteed to give a globally optimal solution, with an example.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: A locally attractive choice can block the optimum; coins {1,3,4} for target 6 are a counterexample.

DAA19's largest-first choice produces 4+1+1, using three coins, while 3+3 uses two. Once the greedy method keeps 4, it cannot repair the earlier choice within its irrevocable policy. Therefore local feasibility and attractiveness do not prove global optimality; a problem-specific exchange or safety proof is needed.

Primary note proof

Study-note section DAA19

> Advantages are often simple decisions and low overhead; limitations are irrevocable errors and problem-specific applicability. A heuristic has no approximation guarantee merely because it is greedy. For coins{1,3,4} and target 6, largest-first gives 4+1+1 while 3+3 uses fewer coins. Activity selection instead has a safe earliest-finish rule for maximizing the number of nonoverlapping unweighted intervals; replacing the first chosen activity in an optimum with an earlier-finishing one leaves at least as much room.

Question sources

Algorithm Design & Analysis review questions · Question 42

Question 128 | NS0510

For ordinary array quicksort, what partition pattern causes quadratic worst-case time, and how can randomized pivot selection or three-way partitioning mitigate relevant workloads? Distinguish expected-time improvements from worst-case and stack-space guarantees.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Repeated very unbalanced partitions cause quadratic work. Random pivots give expected Theta(n log n) time; three-way partitioning helps repeated keys, and smaller-side recursion can limit stack space.

Extreme pivots can leave subproblems of sizes n-1,n-2 and so on, giving quadratic total partition work. Random pivots improve expected time to Theta(n log n) but do not remove quadratic possible executions. Three-way partitioning helps repeated keys, while median-of-three still admits adversarial inputs. Recursing only on the smaller side and iterating on the larger separately bounds stack depth by O(log n). No universal rarity claim is needed.

Primary note proof

Study-note section DAA15

> Balanced splits give $\Theta(n \log n)$; repeatedly highly unbalanced splits give $\Theta(n^2)$. Random pivots give expected $\Theta(n \log n)$, not a deterministic guarantee. Median-of-three does not eliminate adversarial cases; three-way partitioning helps repeated keys. Standard quicksort is unstable, uses expected $\Theta(\log n)$ stack but worst $\Theta(n)$; recursing on the smaller side and iterating on the larger bounds stack by $O(\log n)$.

Question sources

Algorithm Design & Analysis review questions · Question 43

Question 129 | NS0511

For the Job Sequencing algorithm, if two jobs have the same profit, how should ties be broken?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Equal-profit jobs may be processed in any order under the stated unit-job assumptions.

DAA21 explicitly permits arbitrary equal-profit ordering; earlier deadline is an optional tie convention rather than a correctness requirement. Each accepted job still goes into its latest free slot finishing by its deadline. The freedom relies on the one-machine, initially available unit-duration model and does not automatically extend to variable durations or release times.

Primary note proof

Study-note section DAA21

> Sorting plus slot scans costs O(n log n+nD), with D capped at n. Equal-profit order is arbitrary; earlier deadline is optional. Variable durations/release times invalidate this rule.

Question sources

Algorithm Design & Analysis review questions · Question 44

Question 130 | NS0512

Using the note's worst-case bounds, compare matrix-based Prim with Kruskal that comparison-sorts all edges on a dense simple undirected graph, where E=Theta(V^2). Why can this implementation pairing favor Prim asymptotically?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Matrix Prim has O(V^2) time; edge-sorting Kruskal has O(E log E)=O(V^2 log V). The matrix implementation avoids the all-edge sorting factor.

The matrix-based Prim implementation scans vertex connections in O(V^2) time. Kruskal's stated bound includes sorting all E edges, giving O(E log E). For dense simple graphs E has order V^2, so that expression becomes O(V^2 log V), whereas matrix Prim retains its quadratic bound. This explains the advantage of the specified implementations and does not claim that every version of Prim is always faster in measured runtime.

Primary note proof

Study-note section DAA23

> Prim repeatedly adds the cheapest edge crossing from its current tree to an outside vertex. Kruskal sorts edges globally, accepting one only when union-find says its endpoints are in different components. The cut proof exchanges a light crossing edge into some optimum without raising its weight. Matrix Prim costs O(V^2); binary-heap Prim O(E log V), and Kruskal O(E log E). Dense-graph comparisons depend on these implementations. Disconnected input gives a minimum spanning forest.

Question sources

Algorithm Design & Analysis review questions · Question 45

Question 131 | NS0513

Can Dijkstra's Algorithm be applied to a graph with negative edge weights? Why or why not?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Standard Dijkstra is not generally correct with negative edges.

DAA24's example $s \to a=2$, $s \to b=5$, $b \to a=-4$ finalizes a at 2 but a later route improves it to 1. That violates the nonnegative-edge finalization proof even without a negative cycle. Some instances may happen to work, but no general guarantee remains. Bellman-Ford or DAG DP supplies an appropriate alternative under its assumptions.

Primary note proof

Study-note section DAA24

> Nonnegative edges ensure a later path through unsettled vertices cannot improve a finalized distance. With $s \to a=2$, $s \to b=5$, $b \to a=-4$, finalizing a at 2 misses the later distance 1. Binary-heap/list implementation costs $O((V+E)\log V)$, matrix scanning $O(V^2)$. An MST minimizes total connection cost; Dijkstra minimizes each source-path distance, a different objective.

Study-note section DAA27

> For a general graph, Bellman-Ford relaxes all edges for up to $V-1$ passes and tests an additional pass for a reachable negative cycle: $O(VE)$. Thus "single-source shortest path" does not uniquely imply greedy; graph assumptions determine the algorithm.

Question sources

Algorithm Design & Analysis review questions · Question 46

Question 132 | NS0514

What is the time complexity of APSP using the Floyd-Warshall approach and why?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: $\Theta(V^3)$ time and $\Theta(V^2)$ distance storage.

DAA28 runs one intermediate-vertex phase for every k and examines every ordered pair i,j during each phase. V phases times V^2 constant-cost relaxations gives cubic time. The matrix has V^2 entries. These bounds assume bounded-cost arithmetic and do not claim that Floyd is universally optimal among all APSP algorithms or models.

Primary note proof

Study-note section DAA28

> For all-pairs distances initialize D[i, i]=0, D[i, j] to the minimum direct edge weight, and missing edges to∞. Preserve a negative self-loop by taking min(0, its weight). For k outermost, update D[i, j]=min(D[i, j], D[i, k]+D[k, j]); before phase k, intermediates are restricted to earlier vertices. Guard unreachable sums and numeric overflow.

Question sources

Algorithm Design & Analysis review questions · Question 47

Question 133 | NS0515

Explain the difference between 'Divide and Conquer' and 'Dynamic Programming' using the Fibonacci problem.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Naive Fibonacci has overlapping recursive calls; DP reuses them instead of treating them as independent.

DAA26 explicitly says naive Fibonacci is not the independent-subproblem ideal of divide and conquer. Both branches recompute states such as F(n−2), causing exponential calls. A cache or increasing-index table computes each state once, giving linear unit-cost additions. Recursion alone therefore does not establish a well-structured independent divide-and-conquer algorithm.

Primary note proof

Study-note section DAA26

> Naive Fibonacci recomputes F(n−2) across branches: call count is Θ(φ^n), with O(2^n) a valid loose upper bound. F(0)=0, F(1)=1 and F(i)=F(i−1)+F(i−2) yield Θ(n) unit-cost additions with memoization/tabulation. Keeping only two recent values uses Θ(1) machine words; arbitrary-precision output needs growing bit storage. This naive overlapping recursion is not the independent-subproblem ideal of D&C.

Study-note section DAA12

> Divide and conquer solves smaller largely independent instances and combines results.

Question sources

Algorithm Design & Analysis review questions · Question 48

Question 134 | NS0516

What does the recurrence T(n) = 2T(n/2) + O(n) represent and what is its solution?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Two half-size recursive calls plus at most linear combine work; O(n log n), tight Θ(n log n) if combine work is Θ(n).

DAA14 warns that a written +O(n) term alone gives an upper bound rather than a matching lower bound. Expanding logarithmically many levels with at most linear aggregate work gives O(n log n). Ordinary merging actually costs Θ(n), supplying the tight result, but a cheaper unspecified combine function need not do so.

Primary note proof

Study-note section DAA14

> Ordinary array merge sort has T(n)=2T(n/2)+Θ(n)=Θ(n log n) across input orders, Θ(n) auxiliary buffer and Θ(log n) recursion depth. A written +O(n) recurrence alone establishes an upper bound; the tight result also uses linear merge work. Empty/singleton subarrays are base cases.

Question sources

Algorithm Design & Analysis review questions · Question 49

Question 135 | NS0517

How does the Optimal Merge Pattern relate to Huffman Coding?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Both repeatedly combine the two smallest weights; Huffman interprets the resulting tree as a prefix code.

DAA22 uses file sizes as merge weights and frequencies as Huffman weights. Reinsert each combined weight so large values avoid unnecessary depth. In Huffman the weighted path length is ∑frequency×depth, the same merge-tree objective. Equal-weight choices can change codewords without changing the optimal total cost.

Primary note proof

Study-note section DAA22

> Combine the two smallest weights, add their sum to total cost, and reinsert it. A min-heap gives O(n log n) time/O(n) space. Cost p+q counts moved records; worst-case comparisons are p+q−1 for nonempty lists, and actual comparisons may be fewer.

Study-note section DAA22

> Huffman uses frequencies as weights and 0/1 branches to leaf symbols, making a prefix-free code. Cost is∑frequency×depth. An exchange makes the two least frequencies deepest siblings; contract them and induct. Ties can change codes without changing optimal cost. Header overhead can outweigh tiny-file savings.

Question sources

Algorithm Design & Analysis review questions · Question 50

Question 136 | NS0518

In Dynamic Programming, what is 'optimal substructure' and give one example.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: An optimal solution contains suitable optimal subsolutions; a DAG shortest path has an optimal suffix.

DAA25 justifies replacing a subsolution with a better one without breaking feasibility. For DAA27's DAG cost-to-go, if a chosen u→v route used a nonoptimal v-to-target suffix, replacing it with the cheaper suffix would improve the whole route, contradicting optimality. The state must retain enough information for this replacement argument to be valid.

Primary note proof

Study-note section DAA25

> Define state meaning, choices, recurrence, bases and dependency order. Prove replacing a subsolution by a better one improves the full solution; otherwise state information may be missing. Show the recurrence covers all legal final choices and generates feasible candidates, then induct in dependency order. Store choices for reconstruction.

Study-note section DAA27

> For a DAG or multistage graph, compute cost-to-go in reverse topological order: F(t)=0 and F(u)=min over outgoing(u, v) of[w(u, v)+F(v)]. Missing routes have∞. Record the minimizing next vertex. Equivalently propagate source distances forward. Complexity is Θ(V+E) with adjacency lists; negative edges are harmless without cycles. Follow the recurrence's dependencies when choosing the evaluation direction.

Question sources

Algorithm Design & Analysis review questions · Question 51

Question 137 | NS0622

What is the difference between Big O, Big Omega, and Big Theta notation? Provide examples.

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: O is an eventual upper bound, Ω a lower bound, and Θ both.

DAA04 defines the relations by fixed constant-factor inequalities beyond a threshold. For $f=3n^2+4n+1$ and $n\geq 1$, $3n^2\leq f\leq 8n^2$, so $f\in\Theta(n^2)$, and consequently both $O(n^2)$ and $\Omega(n^2)$. It is also $O(n^3)$, showing that upper bounds need not be tight or name an input case.

Primary note proof

Study-note section DAA04

> For eventually nonnegative f, g, $f\in O(g)$ means $f(n)\leq c\cdot g(n)$ for all $n\geq n_0$, with fixed $c>0$. Ω reverses that inequality; Θ requires both bounds. Little-o means $f/g\to 0$, and little-ω means $f/g\to\infty$ where the ratio exists. Θ specifies growth up to constant factors, not exact seconds or identical best/worst cases.

Study-note section DAA04

> Also $3n^2+4n+1\leq 8n^2$ for $n\geq 1$.

Study-note section DAA05

> $O(n^2)$ also implies $O(n^3)$: upper bounds need not be tight.

Question sources

Algorithm Design & Analysis review questions · Question 1

Question 138 | NS0623

Why is Merge Sort $\Theta(n \log n)$ in all cases while Quick Sort can be $O(n^2)$?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Merge has balanced halves with linear merging per level; quicksort can repeatedly keep an n−1 side.

DAA14 yields logarithmically many full linear-work merge levels across input orders. DAA15 allows extreme pivots creating the sum $n+(n-1)+\ldots$ and quadratic work. Random pivots improve expected quicksort performance without a worst-case guarantee. The comparison concerns ordinary merge sorting; optimizations that skip merges require their own best-case analysis.

Primary note proof

Study-note section DAA14

> Ordinary array merge sort has $T(n)=2T(n/2)+\Theta(n)=\Theta(n \log n)$ across input orders, $\Theta(n)$ auxiliary buffer and $\Theta(\log n)$ recursion depth. A written $+O(n)$ recurrence alone establishes an upper bound; the tight result also uses linear merge work. Empty/singleton subarrays are base cases.

Study-note section DAA15

> Balanced splits give $\Theta(n \log n)$; repeatedly highly unbalanced splits give $\Theta(n^2)$. Random pivots give expected $\Theta(n \log n)$, not a deterministic guarantee. Median-of-three does not eliminate adversarial cases; three-way partitioning helps repeated keys. Standard quicksort is unstable, uses expected $\Theta(\log n)$ stack but worst $\Theta(n)$; recursing on the smaller side and iterating on the larger bounds stack by $O(\log n)$.

Question sources

Algorithm Design & Analysis review questions · Question 2

Question 139 | NS0624

When would you choose a greedy algorithm over dynamic programming?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Choose greedy when a local-choice safety proof exists and its lower overhead meets the task.

DAA19 supports exchange arguments that insert the greedy choice into some optimum without worsening it. With that proof, irrevocable decisions can avoid a DP table and simplify implementation. If local safety is absent but a correct reusable-state recurrence exists, DP is a stronger candidate. Optimal substructure alone is not enough to justify greedy.

Primary note proof

Study-note section DAA19

> A greedy algorithm repeatedly selects a locally attractive feasible option without later undoing it. Optimal substructure alone does not prove this safe. An exchange proof inserts the greedy choice into an optimum without worsening cost; recurse. MST cut proofs use this exchange.

Study-note section DAA19

> Advantages are often simple decisions and low overhead; limitations are irrevocable errors and problem-specific applicability.

Question sources

Algorithm Design & Analysis review questions · Question 3

Question 140 | NS0625

Why is naive recursive Fibonacci $O(2^n)$ while DP Fibonacci is O(n)?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: Naive recursion repeats states exponentially; DP computes n states once.

DAA26 gives tight naive call growth Θ(φ^n), so O(2^n) is a valid loose upper bound rather than the exact rate. Memoization or tabulation needs Θ(n) unit-cost additions and can keep two rolling values for value-only output. Arbitrary-precision arithmetic grows with output bit length and is not truly constant cost per addition.

Primary note proof

Study-note section DAA26

> Naive Fibonacci recomputes F(n−2) across branches: call count is Θ(φ^n), with O(2^n) a valid loose upper bound. F(0)=0, F(1)=1 and F(i)=F(i−1)+F(i−2) yield Θ(n) unit-cost additions with memoization/tabulation. Keeping only two recent values uses Θ(1) machine words; arbitrary-precision output needs growing bit storage. This naive overlapping recursion is not the independent-subproblem ideal of D&C.

Question sources

Algorithm Design & Analysis review questions · Question 5

Question 141 | NS0626

Apply the Master Theorem to $T(n) = 4T(n/2) + n^2$. What is the solution?

Hint

State the model and invariant, then derive the result from the cited procedure or recurrence, checking base cases and limitations.

Answer: $\Theta(n^2 \log n)$

DAA10 has a=4, b=2, so n^(log_b a)=n^2. The additive f(n)=n^2 matches that power, selecting the balanced second case. Its logarithmically many levels each contribute $\Theta(n^2)$, including the appropriate base work, so the total is $\Theta(n^2 \log n)$. The polynomial-gap regularity condition belongs to the larger-f third case and is unnecessary here.

Primary note proof

Study-note section DAA10

> For T(n)=aT(n/b)+f(n), a≥1, b>1, let p=log_b a. Level i has a^i nodes of size n/b^i. Compare f with n^p: a polynomial gap below gives Θ(n^p); f=Θ(n^p) gives Θ(n^p log n); a polynomial gap above gives Θ(f(n)) if a f(n/b)≤c f(n) for fixed c<1. The third case needs this regularity condition.

Study-note section DAA10

> Also $4T(n/2)+n^2$ gives $\Theta(n^2 \log n)$, case 2.

Question sources

Algorithm Design & Analysis review questions · Question 6

Introduction to Artificial Intelligence

Questions & Worked Answers

235 questions + 2 source variants

Question 1 | Q00001

Which one of the following is concerned with the meaning of a sentence in knowledge representation?

A. Structure

B. Computational aspect

C. Semantics

D. Syntax

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Semantics

Semantics assigns meaning to sentences; syntax restricts their legal construction (PDF p. 43). The stem asks about meaning rather than representation layout or computational efficiency.

Why each choice is right or wrong

A - Incorrect: Structure concerns arrangement, not the stated meaning rule. See PDF p. 43.

B - Incorrect: Computational aspects concern processing rather than the interpretation definition. See PDF p. 43.

C - Correct: This is precisely the module definition of assigning meaning. See PDF p. 43.

D - Incorrect: Syntax decides legal symbols and sentence construction. See PDF p. 43.

Primary note proof

Study-note section AI24

> Knowledge representation encodes facts, objects, relations, events and rules so a machine can use them. Syntax defines legal expressions; semantics defines their meaning. A knowledge base stores assertions; an inference system derives consequences. Declarative knowledge says what is true; procedural knowledge specifies how to act.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 43 | lines 7-8

> o Semantic also involves assigning a meaning to each sentence.
> o It can be two logics : Propositional Logics and Predicate logics

Question sources

MOE Computer Science Exit Exam · 2016 · Question 1

Question 2 | Q00007

Which one of the following agents works by searching and matching the rules defined by percent and stored internal state?

A. Model-based reflex agent

B. Goal-based agent

C. Simple reflex agent

D. Utility based agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Model-based reflex agent

Taking “percent” as a typographical error for percept, the stored internal state records percept history and the model specifies world evolution (PDF p. 14). That distinguishes model-based reflex selection from a simple current-percept rule.

Why each choice is right or wrong

A - Correct: Combines a model and retained internal state. See PDF p. 14, 14.

B - Incorrect: Goal-based selection needs a desired goal, absent here. See PDF p. 14, 14.

C - Incorrect: Ignores percept history and therefore cannot account for the stated memory. See PDF p. 14, 14.

D - Incorrect: Requires a utility measurement, absent here. See PDF p. 14, 14.

Primary note proof

Study-note section AI06

> A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean. It has no persistent world-state memory. This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

Study-note section AI06

> A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects. It then applies rules to that state. Both kinds can use condition-action rules; memory and a world model distinguish them.

Study-note section AI07

> A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: $EU(a)=\Sigma s\ P(s|a,evidence)U(s)$. Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 14 | lines 29-30

> - Internal State: It is a representation of the current state based on percept history.
> • These agents have the model, "which is knowledge of the world" and based on the model they

Introduction to Artificial Intelligence module | PDF page 14 | lines 17-18

> • The Simple reflex agent works on Condition-action rule, which means it maps the current state to
> action.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 7

Question 3 | Q00028

Assume the entire state space and transition model are known for every task below, despite any hidden current state. Which problem/environment coupling is incorrect?

A. Contingency problem -> partially observable, non-deterministic

B. Multiple state problem deterministic, partially observable

C. Exploration problem -> partially observable, deterministic

D. Single state problem-> fully observable, deterministic

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Exploration problem -> partially observable, deterministic

A known deterministic model with incomplete observation can be handled by a belief state representing several possible current states. Unknown state spaces or transition models create the need for exploration; partial observability alone does not. Nondeterministic partial observation can require contingency branches, while full observation with known deterministic transitions supports a single-state formulation. The explicit knowledge assumption therefore makes the exploration pairing the incorrect one.

Why each choice is right or wrong

A - Incorrect: This is a valid coupling: partial observation and nondeterminism can require contingency branches, so it is not the incorrect pairing.

B - Incorrect: This is valid under the known-model assumption: incomplete observation can require a belief state, so it is not selected.

C - Correct: This is the incorrect coupling: hidden state with a known deterministic model does not by itself require exploration.

D - Incorrect: This is valid: full observation and known deterministic transitions support a single current state.

Primary note proof

Study-note section AI10

> With a known deterministic model and full observation, a single current state can be used. Without observation but with known deterministic transitions, a sensorless or multiple-state problem tracks a belief state: a set of possible states.

Study-note section AI10

> Nondeterminism or partial observation may require a contingency plan with branches based on later percepts. Exploration or online search is required when the state space or transition model is unknown; observation alone does not identify this category. These classifications can overlap. A pairing based only on “partially observable, deterministic” cannot distinguish a known sensorless problem from an unknown exploration problem.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 13 | lines 23-24

> ✓ Deterministic / Non-deterministic − If the next state of the environment is completely determined
> by the current state and the actions of the agent, then the environment is deterministic; otherwise it

Introduction to Artificial Intelligence module | PDF page 13 | lines 14-15

> ✓ Observable / Partially Observable − If it is possible to determine the complete state of the environment at each time point from the percepts it is observable; otherwise it is only partially

Question sources

MOE Computer Science Exit Exam · 2016 · Question 28

Question 4 | Q00033

Which one of the following is correct first order logic for the statement "There is someone standing at the door and is tall"?

A. ∀x (StandingAt (x; door)

B. ∀x (StandingAt (x; door)^ tall(x))

C. ∃x (StandingAt (x; door)^ tall(x))

D. ∃x (StandingAt (x; door) -> toll(x))

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer C: ∃x (StandingAt (x; door)^ tall(x))

“There is someone” requires ∃x, and “standing ... and ... tall” requires StandingAt(x,door) ∧ tall(x), so C asserts one witness with both properties (PDF pp. 46, 49). An implication does not assert its antecedent and therefore cannot assert that anyone is at the door.

Why each choice is right or wrong

A - Incorrect: Incomplete formula with a universal quantifier and no tall predicate. See PDF p. 49, 46.

B - Incorrect: Claims everyone has both properties rather than at least one person. See PDF p. 49, 46.

C - Correct: Existentially binds the same x in both conjuncts. See PDF p. 49, 46.

D - Incorrect: Uses implication instead of conjunction and also spells the predicate toll. See PDF p. 49, 46.

Primary note proof

Study-note section AI26

> Use ∀x(Human(x)→Mortal(x)) for all humans, and ∃x(Student(x)∧Tall(x)) for at least one tall student. These are useful translation patterns, not rules forbidding other connectives. ∀x∃y Likes(x,y) allows a different y for each x; ∃y∀x Likes(x,y) requires one shared y. Negation swaps quantifiers: ¬∀x P(x) ≡ ∃x¬P(x). Only occurrences within the correct quantifier’s scope are bound.

Study-note section AI25

> Propositional atoms represent whole true-or-false claims. ¬P negates; P∧Q requires both; P∨Q requires at least one; P→Q is false only when P is true and Q false; P↔Q requires matching truth values. Thus P→Q ≡ ¬P∨Q. De Morgan gives ¬(P∧Q) ≡ ¬P∨¬Q.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 49 | lines 23-24

> 2. Existential Quantifier: - are the type of quantifiers, which express that the statement within its scope
> is true for at least one instance of something.

Introduction to Artificial Intelligence module | PDF page 46 | lines 14-15

> - Conjunction: A sentence which has ∧ connective such as, P ∧ Q is called a conjunction.
> • Example: Rohan is intelligent and hardworking. It can be written as,

Question sources

MOE Computer Science Exit Exam · 2016 · Question 33

Question 5 | Q00057

Which statements are incorrect? Interpret rational action as maximizing expected performance given available information; do not assume a successful action always exists. Select all that apply.

A. A rational agent know the result of its action, but unrealistic

B. Rational agent take action but not 100% sure of success

C. Rational agent act to achieve utmost success based on information

D. An omniscience agent takes actions for 100% sure of success

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A, D: A rational agent know the result of its action, but unrealistic; An omniscience agent takes actions for 100% sure of success

Rationality concerns choosing the action with the best expected performance given the agent's information. It does not mean knowing actual outcomes in advance, so the first claim incorrectly attributes omniscience to a rational agent. Even omniscience would not ensure success if all available actions fail. The remaining statements correctly allow uncertainty while requiring the best expected choice under the available evidence.

Why each choice is right or wrong

A - Correct: This is false: rationality does not imply knowing actual action outcomes in advance.

B - Incorrect: This is true: a rational action need not guarantee success.

C - Incorrect: This is true when utmost success means best expected performance given the available information.

D - Correct: This is false: complete outcome knowledge cannot guarantee that a successful action exists.

Primary note proof

Study-note section AI04

> A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Study-note section AI04

> Omniscience would mean knowing actual action outcomes in advance. Rationality does not require omniscience or guaranteed success. Even complete outcome knowledge cannot create a successful action when every available action fails. Information gathering and exploration may therefore be rational actions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 12 | lines 20-21

> o It knows the actual effects of its actions and it is impossible in real world.
> → A rational agent behaves according to its precepts and knowledge and attempts to maximize the

Introduction to Artificial Intelligence module | PDF page 12 | lines 22-23

> expected performance.
> 2.4 Structure of Intelligent Agents

Question sources

MOE Computer Science Exit Exam · 2016 · Question 57

Question 6 | Q00059

_____ finds the shortest path to the goal based on cost.

A. Depth first search

B. Iterative deepening

C. Uniform cost search

D. Breath first search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Uniform cost search

Uniform-cost search explicitly seeks the goal path with lowest cumulative cost and uses a priority queue (PDF p. 21). “Shortest” here means minimum cost, under the usual nonnegative-cost setting; minimum depth alone is not sufficient when edge costs vary.

Why each choice is right or wrong

A - Incorrect: Follows depth and is explicitly nonoptimal on p. 19. See PDF p. 21, 21.

B - Incorrect: Increases depth limits, not cumulative-cost priority. See PDF p. 21, 21.

C - Correct: Directly minimizes cumulative path cost. See PDF p. 21, 21.

D - Incorrect: Minimizes steps; equivalent to UCS only when all edge costs are equal. See PDF p. 21, 21.

Primary note proof

Study-note section AI15

UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Study-note section AI15

With finite branching and each step at least ε>0, UCS is complete and optimal. Finite graphs permit nonnegative edges with proper duplicate handling. Dijkstra uses the same minimum-distance principle, often computing all reachable distances; UCS commonly stops at a goal and generates states on demand. Queue initialization is an implementation choice, not a fundamental distinction. Negative edges invalidate the usual finalized-distance argument.

Study-note section AI13

BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI16

DLS runs DFS to depth l and reports cutoff separately from failure. It misses goals deeper than l. Iterative deepening runs DLS with limits 0,1,2,...; for finite b it finds the shallowest solution with O(b^d) time and O(bd) frontier space, optimal for equal step costs. Repeated shallow work is small compared with the deepest level.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 21 | lines 20-21

• The primary goal of the uniform-cost search is to find a path to the goal node which has the lowest
cumulative cost.

Introduction to Artificial Intelligence module | PDF page 21 | lines 26-26

• Uniform cost search is equivalent to BFS algorithm if the path cost of all edges is the same.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 59

Question 7 | Q00172

The data structure used in the standard implementation of Depth First Search is ?

A. Stack

B. Array

C. Linked List

D. Queue

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: Stack

DFS uses a stack (PDF p. 18): the newest pending state is explored first. Arrays or linked lists could implement a stack, but they do not themselves specify the required access order.

Why each choice is right or wrong

A - Correct: Explicitly named implementation structure. See PDF p. 18.

B - Incorrect: May store a stack but does not specify LIFO behavior. See PDF p. 18.

C - Incorrect: May implement a stack but is not the stated abstract structure. See PDF p. 18.

D - Incorrect: FIFO queue is used for BFS on p. 20. See PDF p. 18.

Primary note proof

Study-note section AI14

> DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Study-note section AI13

> BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI-S02

> Binary search exploits the order of a sorted sequence: a comparison narrows the remaining candidate interval instead of exploring arbitrary successor states. Array bisection therefore differs from breadth-first or depth-first traversal of a general state-space graph. A storage container such as an array does not by itself specify FIFO or LIFO frontier behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 18 | lines 34-35

> • DFS uses a stack data structure for its implementation.
> • The process of the DFS algorithm is similar to the BFS algorithm.

Question sources

Addis Ababa University mock exam · Question 74

Question 8 | Q00178

Among the following, which kind of algorithm is used in the Game tree to make decisions of Win/Lose?

A. Heuristic Search Algorithm

B. Min/Max Algorithm

C. Depth First Search Algorithm

D. Greedy Search Algorithm

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: Min/Max Algorithm

Minimax recursively backs up utilities, choosing maxima on MAX turns and minima on MIN turns, so it selects moves against an optimal opponent (PDF pp. 33-34). DFS is its traversal mechanism rather than the adversarial decision rule.

Why each choice is right or wrong

A - Incorrect: A broad category, not the specified alternating-player rule. See PDF p. 33, 34.

B - Correct: Provides the game-tree decision procedure. See PDF p. 33, 34.

C - Incorrect: Traverses the tree but alone does not compare opponent utilities. See PDF p. 33, 34.

D - Incorrect: Chooses local heuristic promise rather than alternating MAX/MIN decisions. See PDF p. 33, 34.

Primary note proof

Study-note section AI22

> Minimax assigns terminal utilities, takes minima at MIN nodes and maxima at MAX nodes, then chooses a maximum-valued root action. It gives the best guaranteed outcome against optimal opposition within the modeled tree. For stochastic outcomes add chance nodes weighted by probabilities.

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 33 | lines 9-10

Mini-max algorithm is a recursive or backtracking algorithm which is used in decision-making and
game theory. It provides an optimal move for the player assuming that opponent is also playing

Introduction to Artificial Intelligence module | PDF page 34 | lines 14-15

• If N is a Max move, take max of successors
• If N is a Min move, take min of successors

Question sources

Addis Ababa University mock exam · Question 80

Computer Science Exit Exam · Sene 2016 · Question 28-2

Computer Science Exit Exam practice collection · Question 205

Question 9 | Q00179

Agents' behavior can be best described by

A. Perception sequence

B. Agent function is performing

C. Sensors and Actuators

D. Environment in which agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Agent function is performing

The module evaluates behavior and performance through the agent function, a mapping from perception history to action (PDF p. 11). B is grammatically incomplete but identifies that function.

Why each choice is right or wrong

A - Incorrect: Input history alone does not specify the selected actions. See PDF p. 11.

B - Correct: Names the function that describes action behavior. See PDF p. 11.

C - Incorrect: Describe input/output mechanisms, not the complete action mapping. See PDF p. 11.

D - Incorrect: Provides context, not the mapping. See PDF p. 11.

Primary note proof

Study-note section AI03

A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in

> which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer's prior instructions.

Study-note section AI03

> An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 11 | lines 12-13

> • The behaviour and performance of intelligent agents have to be evaluated in terms of the agent function.

Question sources

Addis Ababa University mock exam · Question 81

Question 10 | Q00180

Which type of agent deals with happy and unhappy states?

A. Simple reflex agent

B. Model based agent

C. Utility based agent

D. Learning agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer C: Utility based agent

Happy/unhappy is an informal description of desirability. The utility function assigns each state a real number measuring how well actions achieve goals (PDF p. 15). The module does not literally use those emotional terms.

Why each choice is right or wrong

A - Incorrect: Selects current-percept rules, not ranked desirability. See PDF p. 15.

B - Incorrect: Tracks world state, without requiring desirability scores. See PDF p. 15.

C - Correct: Uses the state preference measure described in the stem. See PDF p. 15.

D - Incorrect: Improves through experience rather than defining state desirability. See PDF p. 15.

Primary note proof

Study-note section AI07

A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: EU(a)=Σs P(s | a,evidence)U(s). Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Study-note section AI06

A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean. It has no persistent world-state memory. This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

Study-note section AI08

The performance element selects actions. The learning element changes the agent to improve future performance. The critic evaluates behavior against a performance standard and supplies feedback. The problem generator suggests exploratory actions likely to yield useful experience.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 15 | lines 18-19

• The utility function maps each state to a real number to check how efficiently each action achieves
the goals.

Question sources

Addis Ababa University mock exam · Question 82

Question 11 | Q00181

What is the purpose of "Agent" in Artificial Intelligence?

A. Mapping of goal sequence to an action

B. Work without the direct interference of the people

C. Mapping of environment sequence to an action

D. Mapping of precept sequence to an action

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Mapping of precept sequence to an action

The module defines Agent Function as a map from the percept sequence to an action (PDF p. 10). “Precept” is a spelling error for percept. D matches the definition in the current option order.

Why each choice is right or wrong

A - Incorrect: Goals are not the input sequence in the definition. See PDF p. 10.

B - Incorrect: Describes autonomy, not the function sought. See PDF p. 10.

C - Incorrect: The environment itself is not a percept sequence. See PDF p. 10.

D - Correct: Matches the stated percept-sequence-to-action mapping. See PDF p. 10.

Primary note proof

Study-note section AI03

> A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer’s prior instructions.

Study-note section AI04

> A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 10 | lines 20-21

> • Agent Function − It is a map from the precept sequence to an action.
> • An agent is anything that can perceive its environment through sensors and acts upon that

Question sources

Addis Ababa University mock exam · Question 83

Artificial Intelligence practice questions · Question 25

Question 12 | Q00182

Here “exploratory learning” means discovering structure in data without target labels, oracle labeling or reward-driven interaction. Which learning regime is meant?

A. Supervised learning

B. Active learning

C. Unsupervised learning

D. Reinforcement learning

E. None of the Above

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: Unsupervised learning

Unsupervised learning seeks patterns such as clusters or associations without target labels. Active learning asks an oracle to label selected cases, supervised learning fits labeled input-target examples, and reinforcement learning improves behavior from reward-driven interaction. Specifying the feedback makes the intended use of exploratory precise. The term alone would have several usages, but the stated unlabeled structure-discovery task identifies a definite learning regime.

Why each choice is right or wrong

A - Incorrect: Supervised learning uses labeled targets, excluded by the stem.

B - Incorrect: Active learning obtains oracle labels, excluded by the stem.

C - Correct: Unsupervised learning discovers structure without target labels.

D - Incorrect: Reinforcement learning uses reward-driven interaction, excluded here.

E - Incorrect: The specified regime is present among the choices.

Primary note proof

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI32

> Semi-supervised learning combines some labels with additional unlabeled data. Active learning chooses informative cases for an oracle to label; it still uses labels. “Exploratory learning” is not an exclusive standard synonym: it can mean unsupervised exploration or active data acquisition, so define the feedback. A task named summarization or time-series analysis is not automatically supervised; its training targets determine that.

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 7-8

> ⇒ Unsupervised learning means we are only given the x‘s. In either case, the goal is to estimate f. The system is supplied with a set of training examples consisting only of inputs and is required to

Introduction to Artificial Intelligence module | PDF page 69 | lines 15-16

> ⇒ Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action.

Question sources

Addis Ababa University mock exam · Question 84

Question 13 | Q00183

Let F(x,y,t) mean that person x can fool person y at time t. Over nonempty person and time domains, which statement is equivalent to ∀x∃y∃t ¬F(x,y,t)?

A. Everyone can fool some person at some time

B. Every person has someone whom they cannot fool at any time.

C. No person can fool any person at any time.

D. No one can fool everyone all the time

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer D: No one can fool everyone all the time

For each person x, the formula requires a person y and a time t for which fooling fails. Moving negation across the quantifiers gives ¬∃x∀y∀t F(x,y,t): nobody can fool everyone all the time. The witness and time can depend on x. Requiring failure at every time, or forbidding all fooling events, would be stronger claims.

Why each choice is right or wrong

A - Incorrect: This asserts a successful fooling event for each person, rather than an unsuccessful event.

B - Incorrect: This requires one witness person who cannot be fooled at every time, a stronger time scope than the formula requires.

C - Incorrect: This forbids all fooling events, much stronger than requiring one unsuccessful event for each person.

D - Correct: Negating someone fooling everyone at every time yields, for each person, someone and some time when fooling fails.

Primary note proof

Study-note section AI26

> Use ∀x(Human(x)→Mortal(x)) for all humans, and ∃x(Student(x)∧Tall(x)) for at least one tall student. These are useful translation patterns, not rules forbidding other connectives. ∀x∃y Likes(x,y) allows a different y for each x; ∃y∀x Likes(x,y) requires one shared y. Negation swaps quantifiers: ¬∀x P(x) ≡ ∃x¬P(x). Only occurrences within the correct quantifier's scope are bound.

Study-note section AI44

> The formula ∀x∃y∃t¬F(x,y,t) means that for each person there is a person and time at which fooling fails. By quantifier negation this is ¬∃x∀y∀t F(x,y,t), so nobody fools everybody all the time. Forbidding every successful instance is a much stronger claim than this formula.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 49 | lines 12-13

> 1. Universal Quantifier: - is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing.

Introduction to Artificial Intelligence module | PDF page 49 | lines 23-24

> 2. Existential Quantifier: - are the type of quantifiers, which express that the statement within its scope
> is true for at least one instance of something.

Introduction to Artificial Intelligence module | PDF page 46 | lines 12-13

> - Negation: A sentence such as ¬ P is called negation of P. A literal can be either Positive literal or negative literal.

Question sources

Addis Ababa University mock exam · Question 85

Question 14 | Q00184

A language system must decide which interpretation is intended when a sentence permits more than one meaning or grammatical attachment. Which NLP challenge does this describe?

A. Handling Tokenization

B. Handling Ambiguity of Sentences

C. Handling POS-Tagging

D. Handling Semantics of Sentences

E. All of the mentioned

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer B: Handling Ambiguity of Sentences

The difficulty described is ambiguity: a word or sentence can permit multiple meanings or structural attachments. Tokenization creates units, POS tagging assigns grammatical roles, and semantic analysis concerns meaning. These tasks can help resolve ambiguity, but they are not synonymous with the particular obstacle. The question therefore asks for the named challenge rather than ranking all NLP difficulties.

Why each choice is right or wrong

A - Incorrect: Tokenization decides processing units; it does not name the existence of multiple interpretations.

B - Correct: Ambiguity means that more than one interpretation is possible.

C - Incorrect: POS tagging assigns grammatical categories rather than naming the competing interpretations.

D - Incorrect: Semantics studies meaning generally; the specific obstacle described is multiplicity of interpretations.

E - Incorrect: This includes tokenization, tagging and semantic analysis in general, which do not all name the specific multiple-interpretation obstacle.

Primary note proof

Study-note section AI37

> Tokenization divides text into processing units; tokens may be words, punctuation or subwords. POS tagging assigns grammatical roles; named-entity recognition identifies people, organizations and locations. Parsing identifies grammatical structure; semantics concerns meaning; discourse links sentences; pragmatics uses context and speaker intent. Sentiment analysis predicts evaluative tone. Ambiguity can occur at every level: “bank” has multiple senses, and “I saw the man with a telescope” has multiple attachments.

Study-note section AI37

NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 81 | lines 6-7

- NLP is difficult because Ambiguity and Uncertainty exist in the language.
- Ambiguity

Introduction to Artificial Intelligence module | PDF page 80 | lines 18-19

- Semantic analysis is concerned with the meaning representation. It mainly focuses on the literal meaning of words, phrases, and sentences.

Question sources

Addis Ababa University mock exam · Question 86

Question 15 | Q00199

A goal-directed search strategy uses a problem-specific estimate of remaining cost to order current frontier candidates. Which category is it?

A. Uninformed search

B. Informed search

C. Circular traversal without heuristic estimates

D. Binary search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Informed search

Informed search uses problem-specific heuristic information to estimate remaining cost or promise. Merely retaining the current state would not establish that category, because uninformed search also uses states, successors and known costs. The revised stem supplies the missing estimate. Binary search concerns an ordered interval rather than this state-space frontier, and circular traversal does not identify the heuristic-informed category being tested.

Why each choice is right or wrong

A - Incorrect: Uninformed search lacks the extra remaining-cost heuristic specified here.

B - Correct: Informed search uses problem-specific heuristic estimates to guide the frontier.

C - Incorrect: Traversal without heuristic estimates does not meet the specified remaining-cost-guided strategy.

D - Incorrect: Binary search needs an ordered search interval, not merely goal-directed state exploration.

Primary note proof

Study-note section AI17

A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI-S02

Binary search exploits the order of a sorted sequence: a comparison narrows the remaining candidate interval instead of exploring arbitrary successor states. Array bisection therefore differs from breadth-first or depth-first traversal of a general state-space graph. A storage container such as an array does not by itself specify FIFO or LIFO frontier behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 18 | lines 23-24

• The uninformed search does not contain any domain knowledge such as closeness, the location of
the goal. It is also called level order traversal.

Introduction to Artificial Intelligence module | PDF page 24 | lines 14-15

• It uses domain knowledge. In an informed search, problem information is available which can guide
the search.

Question sources

Mock exam 4 · Question 1

Question 16 | Q00200

An agent’s actions affect later decisions; identical state-action pairs can have different outcomes; and the world changes while the agent deliberates. Which list describes these three properties?

A. Deterministic, fully observable, static

B. Fully observable, sequential, deterministic

C. Partially observable, static, continuous

D. Sequential, non-deterministic, dynamic

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Sequential, non-deterministic, dynamic

The scenario establishes three independent dimensions. Earlier actions affect later decisions, so it is sequential. Outcomes are not fixed by the state-action pair, so it is non-deterministic. Change during deliberation makes it dynamic. It gives no complete observation or continuity specification. Classifying the stated properties avoids an unsupported universal ranking of how difficult different environments must be.

Why each choice is right or wrong

A - Incorrect: Determinism and a static world contradict the uncertain outcomes and ongoing change.

B - Incorrect: Sequential fits, but deterministic outcomes contradict the scenario.

C - Incorrect: A static environment cannot change while the agent deliberates; this list also omits the stated action dependence.

D - Correct: Sequential captures dependent decisions, non-deterministic captures uncertain outcomes, and dynamic captures change during deliberation.

Primary note proof

Study-note section AI09

> Fully observable means sensors reveal relevant state; partially observable means relevant information is hidden or noisy. Deterministic means state and action determine the next state; stochastic means outcomes involve randomness. Strategic uncertainty about an opponent is distinct from dice-like randomness.

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 13 | lines 10-11

> 2.7 Properties of Task environments
> ✓ Discrete / Continuous − If there are a limited number of distinct, clearly defined, states of the

Introduction to Artificial Intelligence module | PDF page 13 | lines 26-27

> ✓ Episodic / Non-episodic − In an episodic environment, each episode consists of the agent perceiving
> and then acting. The quality of its action depends just on the episode itself. Subsequent episodes do

Variant 1 | Alternative version 1

An agent's actions affect later decisions; identical state-action pairs can have different outcomes; and the world changes while the agent deliberates. Which list describes these three properties?

A. Partially observable, non-deterministic, sequential, dynamic

B. Deterministic, sequential (non-episodic), dynamic, continuous

C. Static, episodic, continuous, partially observable

D. Fully observable, discrete, episodic, static

Answer A: Partially observable, non-deterministic, sequential, dynamic

Incomplete observations establish partial observability, uncertain transitions establish non-determinism, consequences for later decisions establish sequentiality, and change during deliberation establishes dynamics. The other lists contradict at least one of those conditions. No universal difficulty comparison is needed.

Why each choice is right or wrong

A - Correct: All four properties match the stated scenario.

B - Incorrect: Determinism contradicts uncertain transition outcomes.

C - Incorrect: Static and episodic contradict ongoing change and consequences for later decisions.

D - Incorrect: Full observability, episodicity and static conditions each contradict the scenario.

Primary note proof

Study-note section AI09

> Fully observable means sensors reveal relevant state; partially observable means relevant information is hidden or noisy. Deterministic means state and action determine the next state; stochastic means outcomes involve randomness. Strategic uncertainty about an opponent is distinct from dice-like randomness.

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Question sources

Mock exam 4 · Question 2

MOE Computer Science Exit Exam · Question 12-4

National Computer Science Exit Exam · Question 64

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 196

Question 17 | Q00201

An agent which mainly operates based on a knowledge about "how things happen in the world" and representation of the current state based on percept history is

A. Simple reflex agent

B. Goal based agent

C. Utility based agent

D. Model based agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Model based agent

The wording reproduces the two model-based factors: knowledge of how the world changes and a current-state representation based on percept history (PDF p. 14). No explicit goal or utility comparison is requested.

Why each choice is right or wrong

A - Incorrect: Ignores history. See PDF p. 14, 14.

B - Incorrect: Adds goals, which are not specified. See PDF p. 14, 14.

C - Incorrect: Adds a utility measure, which is not specified. See PDF p. 14, 14.

D - Correct: Both defining factors match directly. See PDF p. 14, 14.

Primary note proof

Study-note section AI06

> A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects. It then applies rules to that state. Both kinds can use condition-action rules; memory and a world model distinguish them.

Study-note section AI07

> A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: $EU(a)=\Sigma s\ P(s|a,evidence)U(s)$. Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 14 | lines 27-28

- Model: It is knowledge about "how things happen in the world," so it is called a Model-based agent.

Introduction to Artificial Intelligence module | PDF page 14 | lines 29-30

- Internal State: It is a representation of the current state based on percept history.
• These agents have the model, "which is knowledge of the world" and based on the model they

Question sources

Mock exam 4 · Question 3

Question 18 | Q00202

An evaluation technique of classifier accuracy where we randomly partition the data into k mutually exclusive subsets, each approximately equal size and, use Di as test set and others as training set at i-th iteration is _____.

A. Random sampling

B. Cross-validation

C. Stratified cross-validation

D. Holdout method

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: k-fold cross-validation.

AI35 describes exactly the rotation in the stem: build k models, each trained on all folds except one, evaluate on that held-out fold, then average the scores. The stem does not say class proportions are preserved, so stratification is not established. A single holdout split or generic random sampling does not specify this complete rotation.

Why each choice is right or wrong

A - Incorrect: Random sampling alone does not specify the disjoint k-fold rotation of training and test subsets.

B - Correct: The described repeated use of each fold as the held-out subset is k-fold cross-validation.

C - Incorrect: Stratification additionally preserves label proportions, a condition not supplied by equal fold sizes alone.

D - Incorrect: A holdout method uses a fixed split rather than rotating through all k disjoint test folds.

Primary note proof

Study-note section AI35

Train data fits parameters; validation data selects models or settings; a held-out test set estimates final performance. In k-fold validation, train k models, each on k−1 folds and evaluate on the remaining fold, then average scores. Stratification approximately preserves label proportions. Fit preprocessing within each training fold. Time-ordered data needs splits respecting time; repeated tuning on the final test set leaks information.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 2-3

with a set of training examples consisting of inputs and corresponding outputs, and is required to
discover the relation or mapping between then, e.g. as a series of rules, or a neural network.
Agent

Question sources

Mock exam 4 · Question 4

Question 19 | Q00203

An intelligent agent is characterized by

A. Learning

B. Reasoning

C. All of them are answers

D. Perception

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer C: All of them are answers

Perception, reasoning and learning all appear in the intelligent-behavior list (PDF p. 6). C aggregates those three capabilities; it does not imply every individual agent must implement every technique.

Why each choice is right or wrong

A - Incorrect: Listed capability, included in C. See PDF p. 6, 6.

B - Incorrect: Listed capability, included in C. See PDF p. 6, 6.

C - Correct: Collects the three listed capabilities. See PDF p. 6, 6.

D - Incorrect: Listed capability, included in C. See PDF p. 6, 6.

Primary note proof

Study-note section AI01

Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI03

An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 6 | lines 15-16

• Perception involving image recognition and computer vision
• Reasoning

Introduction to Artificial Intelligence module | PDF page 6 | lines 3-5

on the reasoning process. A rational agent is one that acts rationally, that is, is in the best possible
manner.
1.2Typical AI problems

Question sources

Mock exam 4 · Question 5

Question 20 | Q00323

An agent type which has condition-action rule and memory to remind the current state of the world and its evolution is called

A. Simple reflex

B. Model based reflex

C. Goal based

D. Utility based

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Model based reflex

The internal state records percept history; updating it requires how the world evolves and how actions affect it (PDF p. 14). A model-based reflex agent therefore matches the rule-plus-memory description.

Why each choice is right or wrong

A - Incorrect: Uses current percept and ignores the rest of the history. See PDF p. 14, 14.

B - Correct: Has the stated model and internal memory. See PDF p. 14, 14.

C - Incorrect: Would additionally require goal-directed lookahead. See PDF p. 14, 14.

D - Incorrect: Would additionally require utility measurement. See PDF p. 14, 14.

Primary note proof

Study-note section AI06

> A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean. It has no persistent world-state memory. This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

Study-note section AI06

> A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects. It then applies rules to that state. Both kinds can use condition-action rules; memory and a world model distinguish them.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 14 | lines 29-30

> - Internal State: It is a representation of the current state based on percept history.
> • These agents have the model, "which is knowledge of the world" and based on the model they

Introduction to Artificial Intelligence module | PDF page 14 | lines 33-34

> - How the world evolves
> - How the agent's action affects the world.

Question sources

Computer Science model exam · Question 54

University of Gondar model exam · Question 54

Question 21 | Q00324

Assume that you are employed as a machine learning expert at the Ministry of Education (MoE). Which machine learning technique you would apply for the task of describing the existing higher education institutions as first generation, second generation and third generation.

A. Reinforcement Learning

B. Regression

C. Classification

D. Clustering

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: Classification

First, second and third generation are supplied category names. Treating the task as assigning institutions to these predefined labels makes it classification, which the module lists under supervised learning (PDF p. 69). If categories were to be discovered without labels, clustering would instead be relevant.

Why each choice is right or wrong

A - Incorrect: Requires action-dependent rewards, absent here. See PDF p. 69, 69.

B - Incorrect: A regression output is not the category assignment requested. See PDF p. 69, 69.

C - Correct: Matches assignment to the supplied categories. See PDF p. 69, 69.

D - Incorrect: Discovers groupings from inputs rather than assigning known labels. See PDF p. 69, 69.

Primary note proof

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 5-6

> o Classification
> o Regression

Introduction to Artificial Intelligence module | PDF page 69 | lines 10-11

o Most common task is clustering - e.g. taxi agent notices —bad traffic days‖
o Association rule learning

Question sources

Computer Science model exam · Question 55

University of Gondar model exam · Question 55

Question 22 | Q00325

Consider the definition of AI: "The effort to design and implement intelligent software that solves problems by identifying best action sequence to achieve the desired goal" can be views AI as

A. A system that acts humanly

B. A system that thinks humanly

C. A system that acts rationally

D. A system that thinks rationally

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: A system that acts rationally

The definition emphasizes the best action sequence for a goal. The acting-rationally view evaluates how a system acts and performs (PDF p. 6), rather than imitation of people or the internal laws of thought.

Why each choice is right or wrong

A - Incorrect: Would emphasize humanlike external behavior. See PDF p. 6.

B - Incorrect: Would model human thought processes. See PDF p. 6.

C - Correct: Matches goal-directed best possible action. See PDF p. 6.

D - Incorrect: Emphasizes inference and reasoning soundness rather than the action criterion. See PDF p. 6.

Primary note proof

Study-note section AI02

Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic. Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives. Thought processes and observed behavior are different evaluation targets.

Study-note section AI02

The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 6 | lines 2-3

machines that act rationally. The focus is on how the system acts and performs, and not so much on the reasoning process. A rational agent is one that acts rationally, that is, is in the best possible

Question sources

Computer Science model exam · Question 56

University of Gondar model exam · Question 56

Question 23 | Q00326

What is completeness of search algorithms?

A. If there is a difference between the quality of goal states, the agent will find the best one

B. An estimate of how much information the algorithm needs to store for finding a solution

C. An estimate of how many search steps it takes to find a solution

D. Provided that a solution exists, the algorithm will find it

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer D: Provided that a solution exists, the algorithm will find it

Completeness is the guarantee of finding a solution when an appropriately reachable solution exists; the BFS paragraph illustrates it using a finite-depth goal (PDF p. 20). It is distinct from finding the lowest-cost solution and from time or memory requirements.

Why each choice is right or wrong

A - Incorrect: Defines optimality, not completeness. See PDF p. 20, 18.

B - Incorrect: Describes space complexity. See PDF p. 20, 18.

C - Incorrect: Describes time complexity. See PDF p. 20, 18.

D - Correct: Matches solution-finding guarantee. See PDF p. 20, 18.

Primary note proof

Study-note section AI12

Completeness means finding a solution if one exists under stated assumptions; optimality means returning the best solution. Time counts work, often generated nodes; space counts simultaneous storage. Let b be branching factor, d shallowest goal depth, m maximum search depth, and l a depth limit. Graph-search bounds in vertices V and edges E differ from implicit-tree bounds.

Study-note section AI13

BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 20 | lines 26-27

Completeness: BFS is complete, which means if the shallowest goal node is at some finite depth, then
BFS will find a solution.

Introduction to Artificial Intelligence module | PDF page 18 | lines 20-21

Optimal Solution: If a solution has the lowest cost among all solutions.
3.5 Search Strategies

Question sources

Computer Science model exam · Question 57

University of Gondar model exam · Question 57

Question 24 | Q00327

Which one of the view of defining AI concerned with thought processing

A. Viewing AI as a system which has complete autonomous

B. Viewing AI as a system that think either humanly or think rationally

C. Viewing AI as a system that act either humanly or think rationally

D. Viewing AI as a system that act omnisciently to its environment

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Viewing AI as a system that think either humanly or think rationally

Thinking humanly models human cognition; thinking rationally examines ideal inference (PDF p. 5). Both concern thought processing, so B is the complete pairing.

Why each choice is right or wrong

A - Incorrect: Autonomy alone does not specify thought processing. See PDF p. 5, 5.

B - Correct: Includes both thought-oriented approaches. See PDF p. 5, 5.

C - Incorrect: Mixes an action view with a thought view. See PDF p. 5, 5.

D - Incorrect: Omniscience is not one of the four views. See PDF p. 5, 5.

Primary note proof

Study-note section AI02

> Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic. Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives. Thought processes and observed behavior are different evaluation targets.

Study-note section AI04

> Omniscience would mean knowing actual action outcomes in advance. Rationality does not require omniscience or guaranteed success. Even complete outcome knowledge cannot create a successful action when every available action fails. Information gathering and exploration may therefore be rational actions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 5 | lines 5-6

> - Are we looking at the thought process or reasoning ability of the system?
> - Or are we only interested in the final manifestations of the system in terms of its

Introduction to Artificial Intelligence module | PDF page 5 | lines 26-27

> think rationally
> 4. The fourth view of AI is that it is the study of rational agents. This view deals with building

Question sources

Computer Science model exam · Question 58

University of Gondar model exam · Question 58

Question 25 | Q00328

Which one of the following statement is false

A. Turing test is a mechanism to validate AI system that act humanly

B. Validating AI system that act rationally can be made by its performance

C. Validating AI system that thinks humanly is impossible

D. Validating AI system that think rationally can be made by using its reasoning logic

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Validating AI system that thinks humanly is impossible

C makes an absolute impossibility claim. The module describes understanding and modeling human thought as the cognitive-science approach (PDF p. 5), and p. 7 explicitly says computers test theories of the human mind. That contradicts treating validation as inherently impossible.

Why each choice is right or wrong

A - Incorrect: Turing's imitation game tests humanlike behavior. See PDF p. 5, 5, 6.

B - Incorrect: Rational-agent view focuses on actions and performance. See PDF p. 5, 5, 6.

C - Correct: This is the false assertion requested: page7 explicitly says computers can test theories about how the human mind works, contradicting an absolute impossibility claim.

D - Incorrect: This statement accords with the rational-thought view on page5: inference mechanisms, reasoning, soundness and completeness are assessment criteria. It is therefore not the false choice.

Primary note proof

Study-note section AI02

> Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic. Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives. Thought processes and observed behavior are different evaluation targets.

Study-note section AI02

> The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 5 | lines 12-13

> the human thought process. This view is the cognitive science approach to AI. -Think human like 2. The second approach is best embodied by the concept of the Turing Test. Turing held that in future

Introduction to Artificial Intelligence module | PDF page 5 | lines 15-16

> argument Turing put forward the idea of an 'imitation game', in which a human being and a computer would be interrogated under conditions where the interrogator would not know which

Introduction to Artificial Intelligence module | PDF page 6 | lines 2-3

> machines that act rationally. The focus is on how the system acts and performs, and not so much on the reasoning process. A rational agent is one that acts rationally, that is, is in the best possible

Introduction to Artificial Intelligence module | PDF page 7 | lines 9-10

> Cognitive AI: computers are used to test theories about how the human mind works--for example,
> theories about how we recognize faces and other objects, or about how we solve abstract problems.

Introduction to Artificial Intelligence module | PDF page 5 | lines 22-26

> 3. Logic and laws of thought deals with studies of ideal or rational thought process and inference. The emphasis in this case is on the inference mechanism, and its properties. That is how the system
> arrives at a conclusion, or the reasoning behind its selection of actions is very important in this point of view. The soundness and completeness of the inference mechanisms are important here.
> -
> think rationally

Question sources

Computer Science model exam · Question 59

University of Gondar model exam · Question 59

Question 26 | Q00534

Which one of the following is evaluating the degrees of success of an agent?

A. Performance measure

B. Action

C. Knowledge

D. Perception

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Performance measure

The performance measure determines how successful an agent is (PDF p. 10). Actions, knowledge and percepts influence behavior, but evaluating that behavior needs the criterion. In the alternative option order the answer is B.

Why each choice is right or wrong

A - Correct: Explicit success criterion. See PDF p. 10.

B - Incorrect: A selected operation, not its evaluation criterion. See PDF p. 10.

C - Incorrect: Supports decisions but does not define success. See PDF p. 10.

D - Incorrect: Supplies sensed inputs rather than the measure. See PDF p. 10.

Primary note proof

Study-note section AI04

> A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Study-note section AI03

> A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer's prior instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 10 | lines 16-17

> • Performance Measure of Agent -It is the criteria, which determines how successful an agent is.
> • Behavior of Agent – It is the action that agent performs after any given sequence of percepts.

Variant 1 | Alternative version 1

Which one of the following is evaluating the degrees of success of an agent?

A. Its Perception

B. Its Performance measure

C. Its action

D. Its prior kowled

Answer B: Its Performance measure

Apply the same study-note rule to this alternative source set; the choice contrasts below show the derivation.

Why each choice is right or wrong

A - Incorrect: Perception supplies input, not the success criterion (PDF p. 10).

B - Correct: The performance measure determines success (PDF p. 10).

C - Incorrect: An action is assessed by the criterion (PDF p. 10).

D - Incorrect: Prior knowledge helps action selection, but is not the success criterion (PDF p. 12).

Primary note proof

Study-note section AI04

> A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Study-note section AI03

> A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer's prior instructions.

Question sources

Computer Science model exam · 2015 · Question 3

MOE Computer Science Exit Exam · Question 14-4

National Computer Science Exit Exam · Question 65

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 197

Question 27 | Q00541

The component of Expert system which provide interaction between user expert system and Expert system

A. Kowledge base

B. Inference engine

C. User interface

D. Human expert

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: User interface

The user interface explicitly provides interaction between the user and the expert system (PDF p. 64). Knowledge storage and rule application serve different roles.

Why each choice is right or wrong

A - Incorrect: Stores domain knowledge. See PDF p. 64.

B - Incorrect: Applies rules to derive conclusions. See PDF p. 64.

C - Correct: Directly provides user-system interaction. See PDF p. 64.

D - Incorrect: Provides expertise rather than the interaction component. See PDF p. 64.

Primary note proof

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Study-note section AI24

> Knowledge representation encodes facts, objects, relations, events and rules so a machine can use them. Syntax defines legal expressions; semantics defines their meaning. A knowledge base stores assertions; an inference system derives consequences. Declarative knowledge says what is true; procedural knowledge specifies how to act.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 64 | lines 20-21

> • User interface provides interaction between user of the ES and the ES itself.
> • Natural Language Processing so as to be used by the user who is well-versed in the task domain

Question sources

Computer Science model exam · 2015 · Question 10

Question 28 | Q00557

Which Terminology of Natural language processing study about meaning full components of word

A. Phonetics

B. Phonology

C. Morphology

D. Pragmatics

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: Morphology.

AI38 defines morphology through prefixes, stems and suffixes, such as un-happi-ness and cat plus plural -s. These are the meaningful components of words requested by the stem. Phonetics concerns physical sounds, phonology sound patterns, and pragmatics intended meaning in context; none names the study of internal meaningful word parts.

Why each choice is right or wrong

A - Incorrect: Phonetics studies the physical properties of speech sounds rather than meaningful word components.

B - Incorrect: Phonology studies a language's sound patterns and contrasts rather than morpheme structure.

C - Correct: Morphology analyzes meaningful internal parts such as prefixes, stems and suffixes.

D - Incorrect: Pragmatics interprets intended meaning in context, not the internal components of a word.

Primary note proof

Study-note section AI38

> Morphology studies meaningful word parts: un-happi-ness contains a prefix, stem and suffix; cats contains cat and plural -s. It differs from tokenization, which identifies processing boundaries. Phonetics studies physical speech sounds; phonology studies a language's sound patterns and contrasts.

Study-note section AI38

> Prosody concerns stress, rhythm, timing, prominence and intonation across syllables and phrases. A pitch rise can help signal a question; shifting stress can change emphasis. These cues assist dialogue and speech processing. Pragmatics concerns intended meaning in context, not the internal components of words.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 80 | lines 7-8

> 1. Lexical Analysis and Morphological
> - The first phase of NLP is the Lexical Analysis. This phase scans the source code as a

Question sources

Computer Science model exam · 2015 · Question 26

Question 29 | Q00577

The application area of AI which focus on recognize objects based on patterns in the same way as the human visual system does.

A. Expert System

B. Speech recogition

C. Computer vison

D. Robotics

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: Computer vison

Computer vision extracts and interprets image information and explicitly includes object recognition (PDF p. 84). That is the application directly matching visual pattern recognition.

Why each choice is right or wrong

A - Incorrect: Uses domain expertise, not specifically image interpretation. See PDF p. 84, 84.

B - Incorrect: Converts spoken language rather than images. See PDF p. 84, 84.

C - Correct: Directly addresses visual objects. See PDF p. 84, 84.

D - Incorrect: Robotics may use vision but is the broader physical-agent domain. See PDF p. 84, 84.

Primary note proof

Study-note section AI39

> Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 84 | lines 4-5

> o Computer vision automatically extracts, analyzes, and comprehends useful information from a single image or an array of images. This process involves development of algorithms to

Introduction to Artificial Intelligence module | PDF page 84 | lines 20-21

> o Object Recognition – they are installed in supermarkets, cameras, high-end cars such as BMW, GM, and Volvo.

Question sources

Computer Science model exam · 2015 · Question 46

Question 30 | Q00581

From steps of problem solving which one is concerned on looking various sequence of action which lead to the goal state, evaluate it and chose the sequence of optimal action?

A. Execute

B. Goal Formulation

C. problem formulation

D. Search

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer D: Search

The module defines a solution as an action sequence from start to goal and an optimal solution as the lowest-cost such solution (PDF p. 18). Searching is the process of finding/evaluating those sequences, so D matches the activity described.

Why each choice is right or wrong

A - Incorrect: Carries out a selected sequence rather than finding one. See PDF p. 18, 18.

B - Incorrect: Specifies a desired outcome rather than searching paths. See PDF p. 18, 18.

C - Incorrect: Sets up states/actions rather than choosing a path. See PDF p. 18, 18.

D - Correct: Finds the goal-reaching sequence. See PDF p. 18, 18.

Primary note proof

Study-note section AI11

First formulate the goal; then formulate the problem by specifying initial state, legal actions, transition model, goal test, and action costs. Search selects a sequence; execution applies it. A state abstracts away irrelevant world details.

Study-note section AI11

In route finding, states are cities, actions are roads, successors are neighboring cities, and path cost is accumulated distance or time. A solution is a legal path ending at a goal; an optimal solution has minimum cost. Automatic programming or program synthesis searches for a program satisfying a specification; conditional plans and loops require richer representations than a fixed action list.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 18 | lines 19-20

Solution: It is an action sequence which leads from the start node to the goal node.
Optimal Solution: If a solution has the lowest cost among all solutions.

Introduction to Artificial Intelligence module | PDF page 18 | lines 20-21

Optimal Solution: If a solution has the lowest cost among all solutions.
3.5 Search Strategies

Question sources

Computer Science model exam · 2015 · Question 50

Question 31 | Q00600

Which types of rule used in phonology to interpretation of the fluctuation in stress and intonation across the sentences?

A. Phonetic rule

B. Phonemic rule

C. Prosodic rule

D. Morpheme rule

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: Prosodic rules.

AI38 identifies stress and intonation as prosodic phenomena, alongside rhythm, timing and prominence. That directly matches the sentence-level variations in the question. Physical sound production and phonemic contrasts describe other aspects of speech; morphemes are meaningful word parts. The phrase phonology in the stem does not change the specific prosodic level requested.

Why each choice is right or wrong

A - Incorrect: Phonetic rules concern physical speech sounds, while the stated stress and intonation patterns are specifically prosodic.

B - Incorrect: Phonemic contrasts distinguish sound categories; they are not the specific stress-and-intonation rule category requested.

C - Correct: Prosodic rules address stress, prominence, rhythm and intonation across syllables and phrases.

D - Incorrect: Morphemes are meaningful word parts, not the rule category for sentence intonation.

Primary note proof

Study-note section AI38

> Morphology studies meaningful word parts: un-happi-ness contains a prefix, stem and suffix; cats contains cat and plural -s. It differs from tokenization, which identifies processing boundaries. Phonetics studies physical speech sounds; phonology studies a language's sound patterns and contrasts.

Study-note section AI38

> Prosody concerns stress, rhythm, timing, prominence and intonation across syllables and phrases. A pitch rise can help signal a question; shifting stress can change emphasis. These cues assist dialogue and speech processing. Pragmatics concerns intended meaning in context, not the internal components of words.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 79 | lines 2-3

> 7.1 Natural language processing
> NLP stands for Natural Language Processing, which is a part of Computer Science, Human language,

Question sources

Computer Science model exam · 2015 · Question 69

Question 32 | Q00611

A system is evaluated by whether a judge can distinguish its natural-language conversation from a human's. Which of the four AI approaches supplies this evaluation criterion?

A. Thinking humanly

B. Acting Humanly

C. Thinking rationally

D. Acting rationally

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: Acting Humanly

The criterion is observable humanlike behavior: a judge compares the system's conversation with a human's. That is the acting-humanly approach underlying the conversational Turing test. Natural-language ability alone would not establish this classification; the evaluation criterion does. Thinking approaches concern internal processes, while rational approaches judge valid conclusions or effective actions rather than resemblance to humans.

Why each choice is right or wrong

A - Incorrect: Thinking humanly concerns internal mental processes, not only observed conversation.

B - Correct: Acting humanly evaluates humanlike observable behavior, as in the conversational Turing test.

C - Incorrect: Thinking rationally evaluates valid reasoning, not indistinguishability from human dialogue.

D - Incorrect: Acting rationally evaluates expected objective achievement, which need not resemble a human.

Primary note proof

Study-note section AI02

> Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic. Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives. Thought processes and observed behavior are different evaluation targets.

Study-note section AI02

> The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 5 | lines 17-18

> was which, the communication being entirely by textual messages. Turing argued that if the interrogator could not distinguish them by questioning, then it would be unreasonable not to call

Introduction to Artificial Intelligence module | PDF page 6 | lines 10-11

> - Communicating (through natural language).
> - Navigating around obstacles on the streets

Question sources

Computer Science model exam · 2015 · Question 80

Question 33 | Q00619

From which components Intelligence is composed of

A. Reasoning

B. Problem solving

C. Linguistic perception

D. All

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer D: All

Reasoning, learning, language understanding, problem solving and perception are all listed aspects of intelligent behavior (PDF p. 6). D aggregates the capabilities named by the question.

Why each choice is right or wrong

A - Incorrect: One listed capability. See PDF p. 6.

B - Incorrect: Another listed capability. See PDF p. 6.

C - Incorrect: Interpreted as language understanding/perception, both listed. See PDF p. 6.

D - Correct: Includes all three capabilities. See PDF p. 6.

Primary note proof

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 6 | lines 3-6

on the reasoning process. A rational agent is one that acts rationally, that is, is in the best possible
manner.
1.2Typical AI problems
While studying the typical range of tasks that we might expect an —intelligent entity‖ to perform,we

Question sources

Computer Science model exam · 2015 · Question 88

Question 34 | Q00622

An individual who involve in solving problems regarding a certain profession and are considered as knowledge providers in a certain field of study known as ----------

A. Domain Expert

B. Knowledge Engineer

C. User

D. All

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer A: Domain Expert

The knowledge engineer acquires information from a subject expert and organizes it into rules (PDF p. 63). The professional who supplies specialist problem-solving knowledge is therefore the domain expert.

Why each choice is right or wrong

A - Correct: Supplies the subject/domain expertise. See PDF p. 63.

B - Incorrect: Elicits and organizes knowledge from the expert. See PDF p. 63.

C - Incorrect: Uses the system rather than necessarily providing expert knowledge. See PDF p. 63.

D - Incorrect: Does not preserve the distinct roles. See PDF p. 63.

Primary note proof

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 63 | lines 10-11

> o He acquires information from subject expert by recording, interviewing, and observing him at work,
> etc. He then categorizes and organizes the information in a meaningful way, in the form of IF-

Question sources

Computer Science model exam · 2015 · Question 91

Question 35 | Q00623

Which statement best describes a rational artificial agent?

A. An artificial agent that selects actions for the best expected performance given its percept history and knowledge.

B. It used only sensors and effectors to be effective

C. It is a combination of software and hardware agents

D. All

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: An artificial agent that selects actions for the best expected performance given its percept history and knowledge.

A rational artificial agent senses its environment and chooses actions expected to maximize its performance measure given its information. It need not know actual outcomes in advance. Sensors and effectors alone do not supply an action-selection policy, and combining two vaguely named agent categories is not the architecture-plus-program definition. The choice gives the intended rational-agent criterion without imposing omniscience.

Why each choice is right or wrong

A - Correct: This states the rational action-selection criterion without requiring omniscience.

B - Incorrect: Sensors and effectors provide interaction, but alone do not establish intelligent action selection.

C - Incorrect: Architecture plus program is not a combination of a software agent and a hardware agent.

D - Incorrect: The component statements are defective, so their aggregate is not a valid definition.

Primary note proof

Study-note section AI03

> An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Study-note section AI03

> A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer’s prior instructions.

Study-note section AI04

A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 11 | lines 2-3

• An Intelligent Agent must sense, must act, and must be autonomous (to some extent). It also must
be rational.

Introduction to Artificial Intelligence module | PDF page 12 | lines 20-21

o It knows the actual effects of its actions and it is impossible in real world.
→ A rational agent behaves according to its precepts and knowledge and attempts to maximize the

Question sources

Computer Science model exam · 2015 · Question 92

Question 36 | Q00625

Which option gives the complete set of correct descriptions of robotics?

A. It is application area of AI emphasis of creation of intelligent and Efficient robot

B. It is a multi disciplinary area of AI

C. It focus on Electrical, Mechanical and software part of robot

D. All

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer D: All

Robotics integrates mechanical, electrical, control and computing systems to sense, plan and act physically. Building intelligent robots is an AI application, and the required disciplines extend across hardware and software. Thus the three individual descriptions are true. Because this is a single-choice question requesting the complete set, the aggregate is selected and each individual description is an incomplete selection.

Why each choice is right or wrong

A - Incorrect: Creating intelligent, efficient robots is a valid application description, but this alone omits the other true descriptions.

B - Incorrect: Robotics is multidisciplinary, but this alone omits the other true descriptions.

C - Incorrect: Electrical, mechanical and software systems are part of robotics, but this alone omits the other true descriptions.

D - Correct: All three descriptions are compatible with robotics as an integrated discipline that builds physical sensing and acting systems.

Primary note proof

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 82 | lines 10-11

> Robotics is a domain in artificial intelligence that deals with the study of creating intelligent and efficient robots. Robots are the artificial agents acting in real world environment. Robots are aimed at

Introduction to Artificial Intelligence module | PDF page 83 | lines 23-24

> ⇒ Components of robotics
> o Power Supply − The robots are powered by batteries, solar power, hydraulic, or pneumatic

Question sources

Computer Science model exam · 2015 · Question 94

Question 37 | Q00646

Which option describes a conventional computer’s instruction-driven storage and retrieval of information, without making a universal comparison with human intelligence?

A. Machine can figure out complete objects even some part is missed

B. Machines are Naturally intelligent

C. Store and Recall information with set of instruction

D. Take precepts from environment with pattern

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: Store and Recall information with set of instruction

A conventional computer manipulates information according to program instructions, including instructions for storage and retrieval. This describes a mode of computation rather than a universal difference in intelligence between humans and machines. Programs can also implement pattern learning and object recognition. Those capabilities do not change which option names the particular storage-and-retrieval operation requested.

Why each choice is right or wrong

A - Incorrect: Recognizing an occluded object is a perception capability, not the definition of instruction-driven storage and retrieval.

B - Incorrect: A computer's programmed operations do not establish natural biological intelligence.

C - Correct: Storing and recalling information by executing instructions matches the operation described.

D - Incorrect: Processing perceptual patterns is a sensing or recognition task, not specifically storage and retrieval.

Primary note proof

Study-note section AI-S01

> A conventional computer manipulates digital information according to a program or sequence of instructions. Programs can implement storage, retrieval and learning operations. Neural networks can learn pattern recognition from examples, so executing instructions and learning patterns are not mutually exclusive descriptions. This provides no universal claim that every machine is better or worse than every human at completing missing information.

Study-note section AI34

> Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 9 | lines 13-14

> C) Humans recall and store information by patterns rather than algorithms
> D) Machines can‘t think out of the box but humans do

Introduction to Artificial Intelligence module | PDF page 9 | lines 11-12

A) Machines perceive by set of rules rather than patterns
B) Machines can label/figure out a missing part of a given object more effectively than humans

Question sources

Computer Science model exam · 2015 · Question 115

Question 38 | Q00651

Which types of searching strategies is not grouped under blind search?

A. Breadth first search

B. Uniform cost search

C. Constraint satisfaction search

D. Depth limited search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Constraint satisfaction search

BFS, UCS and depth-limited search appear in the uninformed-search sequence (PDF pp. 18-22). Constraint satisfaction is treated separately, so C is the item not grouped under blind search.

Why each choice is right or wrong

A - Incorrect: BFS is in the uninformed sequence. See PDF p. 18, 21, 36.

B - Incorrect: UCS is in the uninformed sequence. See PDF p. 18, 21, 36.

C - Correct: A separately presented constraint-based problem formulation/search topic. See PDF p. 18, 21, 36.

D - Incorrect: Depth-limited search is in the uninformed sequence. See PDF p. 18, 21, 36.

Primary note proof

Study-note section AI21

Backtracking assigns a variable, tries a legal value, and undoes the choice on failure. Choose the variable with fewest remaining values; break ties by degree; try values that rule out fewest neighbor choices. Forward checking removes incompatible neighbor values. Arc consistency removes a value when a binary constraint offers it no support in the other domain. CSP is a problem framework using search and propagation, not a universal fourth uninformed traversal.

Study-note section AI17

A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed

algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI13

BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI16

DLS runs DFS to depth l and reports cutoff separately from failure. It misses goals deeper than l. Iterative deepening runs DLS with limits 0,1,2,...; for finite b it finds the shallowest solution with O(b^d) time and O(bd) frontier space, optimal for equal step costs. Repeated shallow work is small compared with the deepest level.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 18 | lines 22-23

3.5.1 Unformed Search Strategies
• The uninformed search does not contain any domain knowledge such as closeness, the location of

Introduction to Artificial Intelligence module | PDF page 21 | lines 17-18

4. Uniform cost search
• Uniform-cost search is a searching algorithm used for traversing a weighted tree or graph.

Introduction to Artificial Intelligence module | PDF page 36 | lines 7-8

3.7 Constraint Satisfaction Search
Constraint satisfaction is a technique where a problem is solved when its values satisfy certain

Question sources

Computer Science model exam · 2015 · Question 120

Question 39 | Q00655

Which statement correctly distinguishes propositional logic from first-order logic?

A. It evaluates whole propositions using a truth assignment, without quantifying over objects in a domain.

B. It is predicate logic

C. It assigns truth to open predicates without assigning values to their object variables.

D. It contains quantifier

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer A: It evaluates whole propositions using a truth assignment, without quantifying over objects in a domain.

Propositional logic treats atoms as whole true-or-false claims and evaluates compound formulas from a truth assignment. It does not quantify over individual objects. First-order logic adds predicates, object variables, interpretations and quantifiers. The distinction does not mean that propositional truth needs no assignment: the truth values of its atoms still matter.

Why each choice is right or wrong

A - Correct: Propositional evaluation uses atom truth values rather than object-domain quantification.

B - Incorrect: Propositional logic is not identical to first-order predicate logic.

C - Incorrect: Open predicates need interpretation and variable assignments; this is not a propositional truth rule.

D - Incorrect: Object quantifiers belong to first-order logic, not propositional syntax.

Primary note proof

Study-note section AI26

> First-order logic adds objects, predicates, functions and quantified object variables. Human(a) is a property; Likes(a,b) a relation; father(a) a function term. Propositional logic has no object quantifiers, but its truth still depends on the assigned atom values.

Study-note section AI25

> Propositional atoms represent whole true-or-false claims. ¬P negates; P∧Q requires both; P∨Q requires at least one; P→Q is false only when P is true and Q false; P↔Q requires matching truth values. Thus P→Q ≡ ¬P∨Q. De Morgan gives ¬(P∧Q) ≡ ¬P∨¬Q.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 45 | lines 16-17

> A proposition is a declarative statement which is either true or false. It is a technique of knowledge
> representation in logical and mathematical form.

Introduction to Artificial Intelligence module | PDF page 47 | lines 20-21

> • We cannot represent relations like ALL, some, or none with propositional logic. Example:
> • All the girls are intelligent.

Question sources

Computer Science model exam · 2015 · Question 124

Question 40 | Q00658

Which option names the complete set of dataset roles used across model fitting, model selection and final held-out evaluation?

A. Validation dataset

B. Train dataset

C. Test dataset

D. All

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer D: All

Training fits parameters, validation selects models or settings, and a held-out test set estimates final performance. The overall workflow uses all three roles, with the test set kept separate from fitting and repeated tuning. Each individual option is true about one role, but only the aggregate gives the complete set requested by this single-choice question.

Why each choice is right or wrong

A - Incorrect: Validation supports model selection, but alone does not cover fitting and final held-out evaluation.

B - Incorrect: Training fits model parameters, but alone does not cover model selection and final evaluation.

C - Incorrect: Held-out test data estimates final performance, but alone omits fitting and model selection.

D - Correct: Training, validation and test sets together cover the three distinct workflow roles.

Primary note proof

Study-note section AI35

> Train data fits parameters; validation data selects models or settings; a held-out test set estimates final performance. In k-fold validation, train k models, each on k−1 folds and evaluate on the remaining fold, then average scores. Stratification approximately preserves label proportions. Fit preprocessing within each training fold. Time-ordered data needs splits respecting time; repeated tuning on the final test set leaks information.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 2-3

> with a set of training examples consisting of inputs and corresponding outputs, and is required to

discover the relation or mapping between then, e.g. as a series of rules, or a neural network.
Agent

Question sources

Computer Science model exam · 2015 · Question 127

Question 41 | Q00721

algorithm traverses a graph in a Breadth-ward motion and uses a
queue to remember to get the next vertex to start a search, when a dead end occurs in
any iteration.

A. Breadth first search

B. Binary search

C. Depth first search

D. A&C

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: Breadth first search

Expanding a graph breadthwise with a FIFO queue is precisely BFS (PDF pp. 19-20). DFS follows a path deeply and uses a stack; binary search is not this traversal.

Why each choice is right or wrong

A - Correct: Matches breadthwise expansion and FIFO frontier. See PDF p. 19, 20.

B - Incorrect: Not the stated graph traversal. See PDF p. 19, 20.

C - Incorrect: DFS uses a stack and goes deeply first. See PDF p. 19, 20.

D - Incorrect: DFS does not meet the queue/breadthwise description. See PDF p. 19, 20.

Primary note proof

Study-note section AI13

BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Study-note section AI-S02

> Binary search exploits the order of a sorted sequence: a comparison narrows the remaining candidate interval instead of exploring arbitrary successor states. Array bisection therefore differs from breadth-first or depth-first traversal of a general state-space graph. A storage container such as an array does not by itself specify FIFO or LIFO frontier behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 19 | lines 25-26

> • BFS algorithm starts searching from the root node of the tree and expands all successor node at the
> current level before moving to nodes of next level.

Introduction to Artificial Intelligence module | PDF page 20 | lines 3-4

> • Breadth-first search implemented using FIFO queue data structure.
> • If branching factor (average number of child nodes for a given node) = b and depth = d, then number

Question sources

Computer Science model exam 2 · Question 62

Question 42 | Q00889

Which Al approach most directly supports understanding biological intelligence behavior as a scientific goal?

A. Weak AI

B. Applied AI

C. .
Cognitive Al

D. Strong AI

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: .
Cognitive Al

Cognitive AI tests theories of how minds recognize objects and solve problems; the science-based goal is understanding biological intelligence (PDF pp. 7-8). C links these two descriptions.

Why each choice is right or wrong

A - Incorrect: Simulating cognition does not by itself specify the scientific-theory-testing aim. See PDF p. 7, 8.

B - Incorrect: Commercially viable systems are the stated focus of applied AI. See PDF p. 7, 8.

C - Correct: Directly tests theories of mind and cognition. See PDF p. 7, 8.

D - Incorrect: Aims for genuine human-level reasoning rather than specifically theory testing. See PDF p. 7, 8.

Primary note proof

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI01

> Narrow or weak AI addresses bounded tasks, such as face recognition, spam filtering or a voice assistant. General AI denotes broad competence transferable across domains; strong AI also has a philosophical meaning involving actual understanding. Do not equate a humanoid body, autonomous vehicle, or many features with general intelligence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 7 | lines 9-10

> Cognitive AI: computers are used to test theories about how the human mind works--for example,
> theories about how we recognize faces and other objects, or about how we solve abstract problems.

Introduction to Artificial Intelligence module | PDF page 8 | lines 17-19

> Science based AI goal
> • Develop concepts, mechanisms and vocabulary to understand biological intelligent behaviours

Question sources

Computer Science Exit Exam · 2018 · Question 2

Question 43 | Q00895

Why is Greedy Best-First Search not guaranteed to be optimal or complete?

A. It ignores path cost g(n) and relies only on h(n)

B. It always chooses the shallowest node

C. .
It expands nodes in FIFO order

D. It requires full knowledge of the goal state in advance

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: It ignores path cost g(n) and relies only on h(n)

Greedy chooses by estimated remaining cost h(n), ignoring accumulated path cost g(n), so a promising estimate can hide an expensive route. The module also warns of loops and nonoptimality (PDF pp. 25-26). Completeness depends on implementation and state-space assumptions; the general guarantee is absent.

Why each choice is right or wrong

A - Correct: Its h-only choice gives the stated lack of a general guarantee. See PDF p. 25, 26.

B - Incorrect: Shallowest-first describes BFS. See PDF p. 25, 26.

C - Incorrect: FIFO order describes BFS, not heuristic priority. See PDF p. 25, 26.

D - Incorrect: Not the reason the module gives for greedy failures. See PDF p. 25, 26.

Primary note proof

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI18

> A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 25 | lines 21-22

> o Step 3: Remove the node n, from the OPEN list which has the lowest value of h(n), and places it in the CLOSED list.

Introduction to Artificial Intelligence module | PDF page 26 | lines 17-18

> o This algorithm is not optimal.
> • Time Complexity: The worst case time complexity of Greedy best first search is O(bm).

Question sources

Computer Science Exit Exam · 2018 · Question 8

Question 44 | Q00910

Which search strategy tries to expand the node that is closest to the goal, on the grounds that this is likely to lead to a solution quickly; Thus, it evaluates nodes by using just the heuristic function: f (n)=h (n).

A. Uniform-cost search

B. Greedy best-first search

C. ¢.
Depth-first search

D. A* search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Greedy best-first search

Selecting the node estimated closest to the goal uses h(n) alone, as the pure-heuristic and lowest-h OPEN-list rules state (PDF p. 25). The same page prints f(n)=g(n), an internal typo contradicted by those rules; visual inspection confirms the typo is in the PDF.

Why each choice is right or wrong

A - Incorrect: Uses g(n), accumulated cost. See PDF p. 25, 25.

B - Correct: Matches h-only priority. See PDF p. 25, 25.

C - Incorrect: Uses depth priority. See PDF p. 25, 25.

D - Incorrect: Uses g(n)+h(n) rather than h alone. See PDF p. 25, 25.

Primary note proof

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI15

> UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 25 | lines 6-7

> • Pure Heuristic Search - Pure heuristic search is the simplest form of heuristic search algorithms. It
> expands nodes based on their heuristic value h(n).

Introduction to Artificial Intelligence module | PDF page 25 | lines 21-22

> o Step 3: Remove the node n, from the OPEN list which has the lowest value of h(n), and places it in the CLOSED list.

Question sources

Computer Science Exit Exam · 2018 · Question 23

Question 45 | Q00922

Which of the following best defines a heuristic in the context of AI search?

A. A function that estimates how close a state is to the goal

B. A guaranteed optimal path to the goal

C. .
A method that always avoids revisiting states

D. A brute-force enumeration of all possible states

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: A function that estimates how close a state is to the goal

The module says a heuristic estimates how close a state is to the goal (PDF p. 25). An estimate guides node ordering but is neither a path nor a guarantee against revisiting states.

Why each choice is right or wrong

A - Correct: Direct definition. See PDF p. 25.

B - Incorrect: An estimate does not itself specify a guaranteed optimal path. See PDF p. 25.

C - Incorrect: Repeated-state management is a separate search concern. See PDF p. 25.

D - Incorrect: Brute force is the uninformed contrast. See PDF p. 25.

Primary note proof

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI17

Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 25 | lines 2-3

• Heuristic function estimates how close a state is to the goal. It is represented by h(n), and it calculates
the cost of an optimal path between the pair of states. Its value is always positive.

Question sources

Computer Science Exit Exam · 2018 · Question 35

Question 46 | Q00939

Which type of heuristic search strategy that evaluates nodes by combining g (n), the cost to reach the node, and h (n.), the cost to get from the node to the goal: f (n) = g (n) +h (n) Since g(n) gives the path cost from the start node to node n, and h(n) is the estimated cost of the cheapest path from n to the goal, we have f (n) = estimated cost of the cheapest solution through n ?

A. A* search

B. Greedy Best-First Search

C. Uniform-cost search

D. Depth-first search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: A* search

A* uses f(n)=g(n)+h(n), combining the cost already paid with an estimate of the remaining cost. Greedy best-first uses only h; uniform-cost search uses only g; depth-first search follows its stack discipline. The sum is the distinguishing rule. The stray answer annotation attached to the depth-first option has been removed from the practice choices.

Why each choice is right or wrong

A - Correct: A* ranks by accumulated path cost plus estimated remaining cost.

B - Incorrect: Greedy best-first ranks by h alone and omits accumulated cost.

C - Incorrect: Uniform-cost search ranks by g alone and omits the heuristic estimate.

D - Incorrect: DFS follows a stack order rather than minimum g+h.

Primary note proof

Study-note section AI18

A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI17

Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI15

UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 26 | lines 25-26

• A* search is the most commonly known form of best-first search.
• It uses heuristic function h(n), and cost to reach the node n from the start state g(n). It has combined

Introduction to Artificial Intelligence module | PDF page 28 | lines 4-5

the formula f(n)= g(n) + h(n), where g(n) is the cost to reach any node from start state.
• Here we will use OPEN and CLOSED list.

Question sources

Computer Science Exit Exam · 2018 · Question 52

Question 47 | Q00961

Why is perception still a challenge for AI-even though systems can recognize faces?

A. Because perception is unrelated to reasoning

B. Because AI lacks actuators

C. Because open-world inputs may contain occlusion, noise, unfamiliar conditions and incomplete context.

D. Because sensors are too expensive

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: Because open-world inputs may contain occlusion, noise, unfamiliar conditions and incomplete context.

Recognizing faces in some settings does not solve all perception problems. Occlusion can hide relevant features, noise can corrupt observations, and unfamiliar conditions or missing context can make an interpretation unreliable. These are difficulties in interpreting sensory information. They do not require every perception task to use natural-language understanding, and they are distinct from actuator availability or sensor price.

Why each choice is right or wrong

A - Incorrect: Perception can use context and support reasoning; it is not unrelated to it.

B - Incorrect: Actuators implement actions, while the question concerns difficulty interpreting sensory input.

C - Correct: Occlusion, noise and unfamiliar or incomplete context can defeat otherwise successful recognition systems.

D - Incorrect: Cost is not the conceptual reason that open-world interpretation remains difficult.

Primary note proof

Study-note section AI39

> Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Study-note section AI01

> Narrow or weak AI addresses bounded tasks, such as face recognition, spam filtering or a voice assistant. General AI denotes broad competence transferable across domains; strong AI also has a philosophical meaning involving actual understanding. Do not equate a humanoid body, autonomous vehicle, or many features with general intelligence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 7 | lines 13-15

> knowledge. Common sense knowledge is needed to function in complex, open-ended worlds. Such a
> system also needs to understand unconstrained natural language. However these capabilities are not
> yet fully present in today‘s intelligent systems.

Question sources

Computer Science Exit Exam · 2018 · Question 74

Question 48 | Q00970

In which of the following Learning methods an agent receives feedback in the form of rewards or penalties to maximize cumulative reward?

A. Supervised learning

B. Reinforcement learning

C. .
Semi-supervised learning

D. Unsupervised learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Reinforcement learning

Reinforcement learning gives rewards for right actions and penalties for wrong ones, with the goal of obtaining the most reward points (PDF p. 69). B matches both interaction and feedback.

Why each choice is right or wrong

A - Incorrect: Supervised learning uses input-target labels rather than action-reward feedback.

B - Correct: Reinforcement learning improves decisions to maximize cumulative reward from interaction.

C - Incorrect: Semi-supervised learning mixes labeled and unlabeled examples; that mixture does not define cumulative-reward learning.

D - Incorrect: Unsupervised learning discovers unlabeled structure rather than defining an action-reward objective.

Primary note proof

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI32

Semi-supervised learning combines some labels with additional unlabeled data. Active learning chooses informative cases for an oracle to label; it still uses labels. “Exploratory learning” is not an exclusive standard synonym: it can mean unsupervised exploration or active data acquisition, so define the feedback. A task named summarization or time-series analysis is not automatically supervised; its training targets determine that.

Study-note section AI31

Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 15-18

⇒ Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action.
The agent learns automatically with these feedbacks and improves its performance. In reinforcement
learning, the agent interacts with the environment and explores it. The goal of an agent is to get the

Question sources

Computer Science Exit Exam · 2018 · Question 83

Question 49 | Q00995

Which one of the following is wrongly matched as AI environment property pairs?

A. Fully observable <-> Partially observable

B. Discrete <-> Continuous

C. Deterministic <-> Stochastic

D. Static <-> Stochastic

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Static <-> Stochastic

The module pairs static with dynamic, and deterministic with nondeterministic (PDF p. 13). Thus static versus stochastic mixes two independent dimensions; D is the wrong pairing.

Why each choice is right or wrong

A - Incorrect: Opposite observability properties. See PDF p. 13, 13.

B - Incorrect: Opposite state/action granularity properties. See PDF p. 13, 13.

C - Incorrect: Stochastic is the question's term for nondeterministic transition uncertainty. See PDF p. 13, 13.

D - Correct: Incorrect pair: static's counterpart is dynamic. See PDF p. 13, 13.

Primary note proof

Study-note section AI09

> Fully observable means sensors reveal relevant state; partially observable means relevant information is hidden or noisy. Deterministic means state and action determine the next state; stochastic means outcomes involve randomness. Strategic uncertainty about an opponent is distinct from dice-like randomness.

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 13 | lines 17-18

> ✓ Static / Dynamic − If the environment does not change while an agent is acting, then it is static; otherwise it is dynamic.

Introduction to Artificial Intelligence module | PDF page 13 | lines 23-24

> ✓ Deterministic / Non-deterministic − If the next state of the environment is completely determined
> by the current state and the actions of the agent, then the environment is deterministic; otherwise it

Question sources

National Exit Exam practice questions · Question 9

Computer Science Exit Exam practice collection · Question 199

Question 50 | Q01007

Which option describes general data processing without AI learning or inference, rather than naming an AI-related field?

A. Machine learning

B. Robotics

C. Data analysis limited to fixed arithmetic summaries, without learning or inference.

D. Natural language processing

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: Data analysis limited to fixed arithmetic summaries, without learning or inference.

Machine learning develops systems that improve from experience, natural language processing interprets or generates language, and robotics combines sensing, planning and physical action. Data analysis is broader: it may use AI, but fixed arithmetic summaries alone need not involve learning or inference. The option restricts that broad category to ordinary non-AI processing, avoiding the incorrect claim that all data analysis lies outside AI.

Why each choice is right or wrong

A - Incorrect: Machine learning is part of AI, and AI43 explicitly places learning architectures and regimes within it.

B - Incorrect: Robotics combines sensing, planning and physical action and is treated as an AI-related field in AI39.

C - Correct: The restricted fixed-summary task need not use AI; this does not exclude learning-based data analysis.

D - Incorrect: Natural language processing interprets and generates human language and is covered as an AI field in AI37.

Primary note proof

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and

efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Study-note section AI37

NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 6 | lines 12-23

1.3Intelligent behaviour
This discussion brings us back to the question of what constitutes intelligent behavior. Some ofthese
tasks and applications are:
• Perception involving image recognition and computer vision
• Reasoning
• Learning
• Understanding language involving natural language processing, speech processing
• Solving problems
• Robotics
• Thinking
• Acting- in the complex environment
• Knowledge - applying successfully in new situation

Introduction to Artificial Intelligence module | PDF page 67 | lines 29-30

• Data mining and knowledge discovery
• Speech/image/video (pattern) recognition

Question sources

Artificial Intelligence practice questions · Question 1

Question 51 | Q01008

Which techniques can improve learned parameters, candidate solutions or policies using training experience, fitness feedback or reward? Select all that apply. Treat D as a fixed knowledge-based expert system.

A. Artificial neural networks

B. Genetic algorithms

C. Reinforcement learning

D. Expert systems

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer A, B, C: Artificial neural networks; Genetic algorithms; Reinforcement learning

Neural-network training adjusts weights from experience, genetic algorithms evolve candidates using fitness and variation, and reinforcement learning improves policies using rewards from interaction. These are different improvement mechanisms that satisfy the broader capability requested. A fixed expert system applies encoded domain knowledge and does not inherently revise itself through experience. The fixed-system assumption distinguishes using knowledge from learning or optimizing with feedback.

Why each choice is right or wrong

A - Correct: Training adjusts neural weights from examples or feedback and can improve performance.

B - Correct: Genetic search improves a candidate population through fitness-based experience and variation.

C - Correct: RL improves policies from action outcomes and rewards.

D - Incorrect: A conventional fixed expert system applies encoded knowledge without inherently learning from experience.

Primary note proof

Study-note section AI43

> ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 67 | lines 18-19

> important feature of —intelligent‖. Machine learning is the subfield of AI concerned with intelligent
> systems that learn from experience and examples. It is the computational study of algorithms that

Introduction to Artificial Intelligence module | PDF page 69 | lines 15-16

> ⇒ Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action.

Introduction to Artificial Intelligence module | PDF page 71 | lines 10-11

> working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning

Question sources

Artificial Intelligence practice questions · Question 2

Question 52 | Q01009

Which listed programming languages can implement AI or machine-learning algorithms? Select all that apply; this asks about implementation capability, not an exclusive popularity ranking.

A. C++

B. Java

C. Python

D. Ruby

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer A, B, C, D: C++; Java; Python; Ruby

All four languages can implement AI or machine-learning algorithms. The note explicitly identifies Python, C++ and Java as implementation languages. The supplementary primary Rumale project documentation supplies the missing Ruby fact by documenting a Ruby machine-learning library with classifiers and clustering algorithms. This establishes capability rather than a popularity ranking, and avoids treating one widely used language as the only valid implementation choice.

Why each choice is right or wrong

A - Correct: The note explicitly identifies C++ as an AI implementation language.

B - Correct: The note explicitly identifies Java as an AI implementation language.

C - Correct: The note identifies Python as widely used for AI implementation.

D - Correct: Rumale’s primary project documentation demonstrates machine-learning algorithms implemented for Ruby.

Primary note proof

Study-note section AI43

Face recognition is both an applied task and a narrow capability. Broad machine-intelligence goals can include emulating human intelligence and building systems that learn. Humanlike problem-solving alone does not uniquely select expert systems. Python is widely used, but C++ and Java also implement AI; “commonly used” does not isolate one language. McCarthy is associated with naming AI, not its sole invention.

Study-note section AI-S01

AI algorithms are not exclusive to one programming language. Rumale is a Ruby machine-learning library with classification and clustering implementations, demonstrating that Ruby can implement learning algorithms as well as the general-purpose languages already discussed. Language choice and learning regime are different decisions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 8 | lines 21-23

1.10 AI Techniques
Various techniques that have involved can be applied to a variety of AI tasks. The techniques are concerned with how we represent, manipulate and reason with knowledge in order to solve problem.

Question sources

Artificial Intelligence practice questions · Question 3

Question 53 | Q01010

Which listed goal specifically expresses the humanlike approaches to AI, rather than autonomy, general automation or replacement of workers?

A. To replace humans with machines

B. To develop autonomous systems

C. To simulate human intelligence

D. To automate tasks and improve efficiency

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: To simulate human intelligence

Humanlike AI approaches aim to model human thought or produce humanlike behavior. Simulating aspects of human intelligence expresses that goal. Autonomous action, automation and efficiency can also be useful AI objectives, but none requires human imitation. Replacing workers is a social or deployment choice rather than the field’s defining aim. This classification does not impose one primary goal on every AI system.

Why each choice is right or wrong

A - Incorrect: Replacing humans is a deployment choice, not the defining goal of humanlike AI.

B - Incorrect: Autonomy concerns independent action and need not emulate human intelligence.

C - Correct: Simulating human intelligence expresses the humanlike approach to thought or behavior.

D - Incorrect: Automation and efficiency can be achieved without modeling human intelligence.

Primary note proof

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI02

> Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic. Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives. Thought processes and observed behavior are different evaluation targets.

Study-note section AI43

> Face recognition is both an applied task and a narrow capability. Broad machine-intelligence goals can include emulating human intelligence and building systems that learn. Humanlike problem-solving alone does not uniquely select expert systems. Python is widely used, but C++ and Java also implement AI; “commonly used” does not isolate one language. McCarthy is associated with naming AI, not its sole invention.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 8 | lines 9-13

> General AI goal
> • Replicate human intelligent
> • Solve knowledge intensive task
> • Make an intelligent connection between perception and action
> • Enhance human-computer interaction/ communication

Question sources

Artificial Intelligence practice questions · Question 4

Question 54 | Q01011

Which AI application is used to understand and interpret human language?

A. Computer vision

B. Speech recognition

C. Natural language processing

D. Expert systems

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: Natural language processing

NLP is technology for machines to understand, analyze, manipulate and interpret human language (PDF p. 79). Speech recognition is narrower: converting spoken words into text (p. 82).

Why each choice is right or wrong

A - Incorrect: Works with image information. See PDF p. 79.

B - Incorrect: Transcribes spoken words; narrower than language interpretation. See PDF p. 79.

C - Correct: Matches the language-understanding definition. See PDF p. 79.

D - Incorrect: Uses domain expertise, not specifically language interpretation. See PDF p. 79.

Primary note proof

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI39

> Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 79 | lines 4-5

and Artificial Intelligence. It is the technology that is used by machines to understand, analyze, manipulate, and interpret human's languages. It helps developers to organize knowledge for

Question sources

Artificial Intelligence practice questions · Question 5

Question 55 | Q01012

Which technique is defined by a specialized knowledge base of conventional domain facts and rules, with an inference engine applying them to solve domain problems?

A. Genetic algorithms

B. Fuzzy logic

C. Expert systems

D. Reinforcement learning

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Expert systems

An expert system is characterized by specialized domain knowledge and an inference engine that reasons over its knowledge base and working facts. Merely saying that a method uses rules would not distinguish it from fuzzy systems, which may also use rules. Genetic algorithms instead evolve populations, and reinforcement learning uses reward feedback. The architecture in the question supplies the specific distinction.

Why each choice is right or wrong

A - Incorrect: Genetic algorithms evolve candidate populations through selection and variation rather than defining a knowledge-base inference architecture.

B - Incorrect: Fuzzy logic represents graded membership and can use rules, but rules alone do not make it the specialized architecture described.

C - Correct: Expert systems apply domain knowledge through an inference engine operating on a knowledge base and working facts.

D - Incorrect: Reinforcement learning improves decisions from rewards rather than defining the described knowledge-base architecture.

Primary note proof

Study-note section AI40

An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Study-note section AI30

A fuzzy controller fuzzifies measurements, applies graded IF-THEN rules, combines outputs and often defuzzifies to a numerical action. Fuzzy control suits linguistic rules about degrees; reinforcement learning suits reward-based adaptation. Merely saying an environment is complex or uncertain does not uniquely select either method.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 63 | lines 2-3

o It is in the form of IF-THEN-ELSE rules.
o Knowledge Acquisition

Introduction to Artificial Intelligence module | PDF page 63 | lines 20-21

o Applies rules repeatedly to the facts, which are obtained from earlier rule application.
o Adds new knowledge into the knowledge base if required.

Question sources

Artificial Intelligence practice questions · Question 6

Question 56 | Q01013

Which learning regime discovers structure in unlabeled data without action-reward interaction? Select the regime rather than a neural architecture family.

A. Supervised learning

B. Unsupervised learning

C. Reinforcement learning

D. Deep learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Unsupervised learning

Unsupervised learning seeks clusters, associations or other structure without target labels. Reinforcement learning also need not receive correct-action labels, but it learns through actions and rewards, which this scenario excludes. Deep learning describes an architecture family that can use several learning regimes. Supervised learning requires labeled targets. Naming both the feedback and the requested category therefore isolates the intended regime.

Why each choice is right or wrong

A - Incorrect: Supervised learning requires labeled input-target examples.

B - Correct: Unsupervised learning discovers structure without target labels.

C - Incorrect: RL learns from action rewards, excluded by the stem.

D - Incorrect: Deep learning is an architecture family, not the specific unlabeled structure-discovery regime requested.

Primary note proof

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI43

> ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 7-8

> ⇒ Unsupervised learning means we are only given the x‘s. In either case, the goal is to estimate f. The system is supplied with a set of training examples consisting only of inputs and is required to

Introduction to Artificial Intelligence module | PDF page 76 | lines 7-8

> means there is no teacher as in supervised learning. After receiving the feedback, the network performs adjustments of the weights to get better critic information in future.

Question sources

Artificial Intelligence practice questions · Question 7

Question 57 | Q01014

What does the Turing test assess?

A. Machine learning capabilities

B. Expert system knowledge

C. Natural language processing proficiency

D. Artificial intelligence in simulating human conversation

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer D: Artificial intelligence in simulating human conversation

Turing's test asks whether an interrogator can distinguish a computer from a human using textual questions (PDF p. 5). D captures the overall conversational behavioral test; NLP proficiency is a component rather than the full criterion.

Why each choice is right or wrong

A - Incorrect: Not a test specifically of learning. See PDF p. 5.

B - Incorrect: Not restricted to expert-system domain knowledge. See PDF p. 5.

C - Incorrect: A required conversational ability, not the complete assessment. See PDF p. 5.

D - Correct: Matches indistinguishability in textual human conversation. See PDF p. 5.

Primary note proof

Study-note section AI02

> The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 5 | lines 18-20

> interrogator could not distinguish them by questioning, then it would be unreasonable not to call the computer intelligent. Turing's 'imitation game' is now usually called 'the Turing test' for intelligence.

Question sources

Artificial Intelligence practice questions · Question 8

Question 58 | Q01015

Which AI technique aims to mimic the structure and function of the human brain?

A. Genetic algorithms

B. Neural networks

C. Expert systems

D. Reinforcement learning

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Neural networks

Artificial neural networks are brain-inspired abstractions built from connected processing units with adjustable weights. This is inspiration rather than a literal recreation of the brain. Genetic algorithms use evolutionary population mechanisms, expert systems use encoded domain knowledge, and reinforcement learning defines a feedback regime that can be implemented with many model types, including neural networks.

Why each choice is right or wrong

A - Incorrect: Genetic algorithms are inspired by population selection and genetic variation, not a network of neural processing units.

B - Correct: Artificial neural networks are brain-inspired abstractions of interconnected weighted processing units.

C - Incorrect: Expert systems encode domain knowledge for inference rather than defining a brain-inspired network structure.

D - Incorrect: Reinforcement learning defines reward-based interaction and can use neural or non-neural implementations.

Primary note proof

Study-note section AI34

> A neuron computes z=Σiwixi+b and output a=φ(z). For x=(2,1), w=(0.5,−1), b=0.2, z=0.2. Linear activation returns z; sigmoid returns 1/(1+e^(−z)); tanh is bipolar; ReLU returns max(0,z). Nonlinearities let multiple layers represent nonlinear relationships.

Study-note section AI20

> A genetic algorithm maintains a population, evaluates fitness, selects parents, applies crossover and mutation, and repeats replacement. It searches without guaranteeing global optimality. Means-ends analysis selects a difference between current and goal states, chooses an operator

reducing it, and recursively solves its preconditions. Neither method is simply A* running backward.

Study-note section AI40

An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI34

Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 71 | lines 6-8

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way
biological nervous systems, such as the brain, process information. The key element of this paradigm
is the novel structure of the information processing system.

Question sources

Artificial Intelligence practice questions · Question 9

Question 59 | Q01016

Which AI application is used to make predictions based on historical data?

A. Data mining

B. Pattern recognition

C. Predictive analytics

D. Genetic algorithms

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Predictive analytics.

AI31 directly defines predictive analytics as using historical patterns to forecast outcomes. This names the application requested by the question. Data mining and pattern recognition are broader related activities that can contribute, while genetic algorithms perform evolutionary optimization. Their possible supporting roles do not make them the precise application term defined in the note.

Why each choice is right or wrong

A - Incorrect: Data mining is a broader discovery activity; the note names historical-outcome forecasting specifically as predictive analytics.

B - Incorrect: Recognizing patterns can support prediction, but does not name the defined historical-forecasting application as precisely.

C - Correct: Predictive analytics is explicitly defined as forecasting outcomes using historical patterns.

D - Incorrect: Genetic algorithms evolve candidate solutions through fitness and variation, rather than naming historical forecasting itself.

Primary note proof

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI20

> A genetic algorithm maintains a population, evaluates fitness, selects parents, applies crossover and mutation, and repeats replacement. It searches without guaranteeing global optimality. Means-ends analysis selects a difference between current and goal states, chooses an operator reducing it, and recursively solves its preconditions. Neither method is simply A* running backward.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 68 | lines 21-22

> • Machine learning constructs or uses the algorithms that learn from historical data.
> • Supervised

Introduction to Artificial Intelligence module | PDF page 71 | lines 18-19

> of information it has been given to analyse. This expert can then be used to provide projections given
> new situations of interest and answer "what if" questions. Other advantages include:

Question sources

Artificial Intelligence practice questions · Question 10

Question 60 | Q01017

Which descriptions explicitly establish a bounded task-specific AI capability, rather than only a body shape or an unspecified “general-purpose” label? Select all that apply.

A. Autonomous vehicles limited to driving tasks.

B. Virtual assistants like Siri or Alexa performing bounded assistant tasks.

C. Humanoid robots

D. Robots described only as general-purpose, with no capability details.

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer A, B: Autonomous vehicles limited to driving tasks.; Virtual assistants like Siri or Alexa performing bounded assistant tasks.

Narrow AI addresses bounded tasks rather than broad competence transferable across domains. Driving-only automation and a bounded voice assistant therefore establish task-specific capabilities. A humanoid body reveals shape, not intellectual scope, while an unexplained general-purpose label supplies no demonstrated cross-domain competence. The question asks which descriptions establish narrowness, so the two insufficient descriptions are excluded without claiming that their particular robots could never be narrow.

Why each choice is right or wrong

A - Correct: Autonomous driving is a bounded capability and does not by itself establish broad cross-domain general intelligence.

B - Correct: Virtual voice assistants are the note’s direct examples of bounded narrow-AI capability.

C - Incorrect: Humanoid shape alone does not establish a bounded task-specific capability.

D - Incorrect: An undefined general-purpose label does not establish a specific bounded capability.

Primary note proof

Study-note section AI01

> Narrow or weak AI addresses bounded tasks, such as face recognition, spam filtering or a voice assistant. General AI denotes broad competence transferable across domains; strong AI also has a philosophical meaning involving actual understanding. Do not equate a humanoid body, autonomous vehicle, or many features with general intelligence.

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 7 | lines 3-5

> Weak AI: deals with the creation of some form of computer-based artificial intelligence that cannot
> truly reason and solve problems, but can act as if it were intelligent. Weak AI holds that suitably programmed machines can simulate human cognition.

Introduction to Artificial Intelligence module | PDF page 7 | lines 19-20

> • In Robotics, we have been able to make vehicles that are mostly autonomous.
> • In Natural language processing, we have systems that are capable of simple machine

Question sources

Artificial Intelligence practice questions · Question 11

Question 61 | Q01018

Which learning regime is defined by an agent choosing actions and improving its policy from environmental rewards or penalties, rather than by a model architecture?

A. Reinforcement learning

B. Supervised learning

C. Unsupervised learning

D. Deep learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer A: Reinforcement learning

Reinforcement learning is defined by interaction: the agent acts, receives reward feedback and improves its policy to increase expected cumulative return. Correct-action target labels are not required. Supervised and unsupervised learning use different feedback definitions. Deep learning describes a model architecture and may support reinforcement learning, but it does not name the particular feedback regime asked for.

Why each choice is right or wrong

A - Correct: Reinforcement learning uses action-reward interaction to improve expected return.

B - Incorrect: Supervised learning uses supplied input-target pairs rather than this reward-feedback definition.

C - Incorrect: Unsupervised learning seeks unlabeled structure rather than an action-reward policy.

D - Incorrect: Deep learning specifies multilayer neural representations and can implement several feedback regimes.

Primary note proof

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI43

> ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples.

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 16-18

> reward for each right action and gets a penalty for each wrong action.
> The agent learns automatically with these feedbacks and improves its performance. In reinforcement

learning, the agent interacts with the environment and explores it. The goal of an agent is to get the

Question sources

Artificial Intelligence practice questions · Question 12

Question 62 | Q01019

Which AI application is used to identify and classify objects in images?

A. Natural language processing

B. Speech recognition

C. Computer vision

D. Expert systems

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: Computer vision

Computer vision automatically analyzes image information and includes object recognition (PDF p. 84), directly matching identifying/classifying image objects.

Why each choice is right or wrong

A - Incorrect: Processes language rather than image objects. See PDF p. 84, 84.

B - Incorrect: Transcribes speech. See PDF p. 84, 84.

C - Correct: The explicitly visual recognition application. See PDF p. 84, 84.

D - Incorrect: Domain expertise alone does not specify vision. See PDF p. 84, 84.

Primary note proof

Study-note section AI39

Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Study-note section AI37

NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI40

An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 84 | lines 4-5

o Computer vision automatically extracts, analyzes, and comprehends useful information from a single image or an array of images. This process involves development of algorithms to

Introduction to Artificial Intelligence module | PDF page 84 | lines 20-21

o Object Recognition – they are installed in supermarkets, cameras, high-end cars such as BMW, GM, and Volvo.

Question sources

Artificial Intelligence practice questions · Question 13

Question 63 | Q01020

Which approach represents imprecise linguistic categories such as warm by graded membership between zero and one, rather than treating membership as strictly true or false?

A. Genetic algorithms

B. Fuzzy logic

C. Neural networks

D. Reinforcement learning

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Fuzzy logic

Fuzzy logic represents imprecise categories through graded membership, so an input can belong to a category to a degree rather than only completely or not at all. This differs from uncertainty about whether an event occurs. Genetic search, neural networks and reinforcement learning can be combined with fuzzy methods, but their defining mechanisms do not supply this membership interpretation.

Why each choice is right or wrong

A - Incorrect: Genetic algorithms evolve populations; graded category membership is not their defining mechanism.

B - Correct: Fuzzy logic represents degrees of membership in vague categories.

C - Incorrect: Neural networks adjust weighted representations and do not inherently define fuzzy membership.

D - Incorrect: Reinforcement learning uses rewards and policy improvement rather than defining graded category membership.

Primary note proof

Study-note section AI30

> Fuzzy sets assign graded membership μ(x) between 0 and 1 to concepts such as warm or tall. A temperature can belong to warm with degree 0.7. This is not a 70% probability that an otherwise crisp warm proposition is true. Boolean logic uses only true and false; probability describes uncertainty about events.

Study-note section AI20

> A genetic algorithm maintains a population, evaluates fitness, selects parents, applies crossover and mutation, and repeats replacement. It searches without guaranteeing global optimality. Means-ends analysis selects a difference between current and goal states, chooses an operator reducing it, and recursively solves its preconditions. Neither method is simply A* running backward.

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI34

> Training adjusts weights, often through gradients and backpropagation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 55 | lines 27-29

> • Probabilistic reasoning is a way of knowledge representation where we apply the concept of probability to indicate the uncertainty in knowledge.
> • In probabilistic reasoning, we combine probability theory with logic to handle the uncertainty.

Introduction to Artificial Intelligence module | PDF page 71 | lines 15-16

> Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data,
> can be used to extract patterns and detect trends that are too complex to be noticed by either humans or

Question sources

Artificial Intelligence practice questions · Question 14

Question 64 | Q01021

Which AI application is used to generate human-like text or speech?

A. Natural language processing

B. Speech recognition

C. Computer vision

D. Expert systems

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer A: Natural language processing

NLG is the NLP component for generating language from computerized representations, including text and sentence planning (PDF pp. 79-80). Thus NLP is the offered application matching generation; the source does not provide a full speech-synthesis treatment.

Why each choice is right or wrong

A - Correct: Includes the explicit language-generation component. See PDF p. 79, 80.

B - Incorrect: Recognizes spoken input rather than generating output. See PDF p. 79, 80.

C - Incorrect: Works with image understanding. See PDF p. 79, 80.

D - Incorrect: Generates domain conclusions, not necessarily natural-language output. See PDF p. 79, 80.

Primary note proof

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI39

> Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 79 | lines 31-33

> - Natural Language Generation (NLG) acts as a translator that converts the computerized data into natural language representation. It mainly involves Text planning, Sentence planning, and Text Realization.

Introduction to Artificial Intelligence module | PDF page 80 | lines 3-4

> - NLG is the process of writing or generating language.
> - NLG - It produces non-linguistic outputs from natural language inputs.

Question sources

Artificial Intelligence practice questions · Question 15

Question 65 | Q01022

Which technique involves evolving solutions using principles from genetics and natural selection?

A. Genetic algorithms

B. Neural networks

C. Reinforcement learning

D. Expert systems

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: Genetic algorithms

AI20 describes a genetic algorithm as maintaining a population, evaluating fitness, selecting parents and applying crossover and mutation. Those operations embody the genetics and natural-selection principles in the stem. Neural networks, reward-based learning and rule-based expert systems use different organizing mechanisms, even where hybrid systems combine them with evolutionary search.

Why each choice is right or wrong

A - Correct: Genetic algorithms use fitness, parent selection, crossover and mutation to evolve candidate populations.

B - Incorrect: Neural networks learn weighted representations; evolutionary population replacement is not required.

C - Incorrect: Reinforcement learning improves policies from environmental rewards rather than requiring genetic variation.

D - Incorrect: Expert systems reason with specialized encoded knowledge rather than requiring evolutionary populations.

Primary note proof

Study-note section AI20

A genetic algorithm maintains a population, evaluates fitness, selects parents, applies crossover and mutation, and repeats replacement. It searches without guaranteeing global optimality. Means-ends analysis selects a difference between current and goal states, chooses an operator reducing it, and recursively solves its preconditions. Neither method is simply A* running backward.

Study-note section AI34

A neuron computes z=Σiwixi+b and output a=φ(z). For x=(2,1), w=(0.5,−1), b=0.2, z=0.2. Linear activation returns z; sigmoid returns 1/(1+e^(−z)); tanh is bipolar; ReLU returns max(0,z). Nonlinearities let multiple layers represent nonlinear relationships.

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI40

An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 8 | lines 36-37

o Genetic Algorithms
1. Which one of the following is correctly matched with the definition and approaches of AI

Question sources

Artificial Intelligence practice questions · Question 16

Question 66 | Q01023

Which technique directly represents degrees of linguistic concepts such as warm and applies graded IF-THEN rules to choose an action?

A. Fuzzy logic

B. Reinforcement learning

C. Expert systems

D. Genetic algorithms

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer A: Fuzzy logic

Fuzzy control represents degrees of membership in concepts such as warm and combines graded rules to produce an action. This handles vague category boundaries rather than requiring probabilistic uncertainty or reward-based adaptation. Reinforcement learning improves policies through rewards, expert systems apply specialized knowledge, and genetic algorithms evolve populations. The added representation and rule requirements distinguish fuzzy control from other methods that may also address complex decisions.

Why each choice is right or wrong

A - Correct: Fuzzy logic uses graded membership and fuzzy IF-THEN rules.

B - Incorrect: RL uses reward-driven adaptation rather than defining graded linguistic membership.

C - Incorrect: An ordinary expert system need not represent degrees of membership in linguistic categories.

D - Incorrect: Genetic algorithms evolve candidates through fitness and variation rather than fuzzy rule evaluation.

Primary note proof

Study-note section AI30

> Fuzzy sets assign graded membership μ(x) between 0 and 1 to concepts such as warm or tall. A temperature can belong to warm with degree 0.7. This is not a 70% probability that an otherwise crisp warm proposition is true. Boolean logic uses only true and false; probability describes uncertainty about events.

Study-note section AI30

> A fuzzy controller fuzzifies measurements, applies graded IF-THEN rules, combines outputs and often defuzzifies to a numerical action. Fuzzy control suits linguistic rules about degrees; reinforcement learning suits reward-based adaptation. Merely saying an environment is complex or uncertain does not uniquely select either method.

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 12 | lines 22-23

> expected performance.
> 2.4 Structure of Intelligent Agents

Introduction to Artificial Intelligence module | PDF page 69 | lines 18-19

> learning, the agent interacts with the environment and explores it. The goal of an agent is to get the
> most reward points, and hence, it improves its performance. The robotic dog, which automatically

Introduction to Artificial Intelligence module | PDF page 60 | lines 21-22

> • Instructing and assisting human in decision
> making

Question sources

Artificial Intelligence practice questions · Question 17

Question 67 | Q01024

Which AI application is used to simulate human conversation?

A. Natural language processing

B. Speech recognition

C. Computer vision

D. Expert systems

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer A: Natural language processing

The module lists chatbots as an important NLP application used for chat services (PDF p. 82). That directly supports NLP for simulated conversation.

Why each choice is right or wrong

A - Correct: Includes the chatbot application. See PDF p. 82.

B - Incorrect: Transcription alone is not conversation. See PDF p. 82.

C - Incorrect: Image comprehension does not define conversation. See PDF p. 82.

D - Incorrect: Expert conclusions may be communicated, but chat is specifically listed under NLP. See PDF p. 82.

Primary note proof

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI02

> The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 82 | lines 5-6

> - Chatbot - is one of the important applications of NLP. It is used by many companies to provide the customer's chat services.

Question sources

Artificial Intelligence practice questions · Question 18

Question 68 | Q01025

A collection of documents has no target labels. The task is to group similar documents, with no action-reward interaction. Which learning regime describes this objective, regardless of the architecture used?

A. Supervised learning

B. Unsupervised learning

C. Reinforcement learning

D. Deep learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Unsupervised learning

Grouping similar documents without target labels is clustering, an unsupervised learning objective. The data modality does not determine the feedback regime, and a neural architecture could still be used for this task. Supervised learning requires targets; reinforcement learning requires an action-and-reward objective. The question asks for the learning regime rather than which architecture might process documents.

Why each choice is right or wrong

A - Incorrect: Supervised learning requires target labels, absent from the described task.

B - Correct: Unsupervised learning discovers structure, including clusters, without supplied targets.

C - Incorrect: No actions or reward feedback are present, so this is not the reinforcement objective.

D - Incorrect: Deep learning describes a multilayer architecture that might implement clustering, not the learning regime.

Primary note proof

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI43

> ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples.

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 7-10

> ⇒ Unsupervised learning means we are only given the x‘s. In either case, the goal is to estimate f. The system is supplied with a set of training examples consisting only of inputs and is required to

discover for itself what appropriate outputs should be,
o Most common task is clustering - e.g. taxi agent notices —bad traffic days‖

Question sources

Artificial Intelligence practice questions · Question 19

Question 69 | Q01026

Which AI technique involves training a model on a large dataset and multiple layers of artificial neurons?

A. Genetic algorithms

B. Neural networks

C. Expert systems

D. Reinforcement learning

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Neural networks

AI34 describes artificial neurons with learned weights and nonlinear outputs, then defines deep learning through multiple representation layers. This matches the large-data, multilayer neural architecture in the stem. Reinforcement learning defines reward-based feedback rather than a specific architecture; genetic algorithms and expert systems do not inherently consist of layered artificial neurons.

Why each choice is right or wrong

A - Incorrect: Genetic algorithms evolve populations and are not defined by layered artificial neurons.

B - Correct: A layered collection of artificial neurons is a neural-network architecture whose weights are adjusted during training.

C - Incorrect: Expert systems define knowledge-base reasoning rather than layers of trainable neurons.

D - Incorrect: Reinforcement learning specifies the reward objective, not a required layered-neuron architecture.

Primary note proof

Study-note section AI34

A neuron computes $z=\Sigma iwixi+b$ and output $a=\varphi(z)$. For $x=(2,1)$, $w=(0.5,-1)$, $b=0.2$, $z=0.2$. Linear activation returns z; sigmoid returns $1/(1+e^{(-z)})$; tanh is bipolar; ReLU returns $\max(0,z)$. Nonlinearities let multiple layers represent nonlinear relationships.

Study-note section AI20

A genetic algorithm maintains a population, evaluates fitness, selects parents, applies crossover and mutation, and repeats replacement. It searches without guaranteeing global optimality. Means-ends analysis selects a difference between current and goal states, chooses an operator reducing it, and recursively solves its preconditions. Neither method is simply A* running backward.

Study-note section AI40

An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI34

Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 71 | lines 9-10

It is composed of large number of highly interconnected simple processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is

Introduction to Artificial Intelligence module | PDF page 74 | lines 4-6

- Network Topology
- Adjustments of Weights or Learning
- Activation Functions

Question sources

Artificial Intelligence practice questions · Question 20

Question 70 | Q01027

What is Artificial Intelligence?

A. Artificial Intelligence is a field that aims to make humans more intelligent

B. Artificial Intelligence is a field that aims to improve the security

C. Artificial Intelligence is a field that aims to develop intelligent machines

D. Artificial Intelligence is a field that aims to mine the data

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Artificial Intelligence is a field that aims to develop intelligent machines

The module defines AI as designing intelligence in an artificial device (PDF p. 4). Developing intelligent machines is the matching general description, rather than an individual application.

Why each choice is right or wrong

A - Incorrect: Making people smarter is not the stated field definition. See PDF p. 4.

B - Incorrect: Security is a possible application, not the field definition. See PDF p. 4.

C - Correct: Matches intelligence in an artificial device. See PDF p. 4.

D - Incorrect: Data mining is one learning application, not the field's definition. See PDF p. 4.

Primary note proof

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 4 | lines 26-27

> is concerned with the design of intelligence in an artificial device. The term was coined by McCarthy
> in 1956. There are two ideas in the definition. i.e Intelligence, and Artificial device.

Question sources

Artificial Intelligence practice questions · Question 21

Question 71 | Q01028

Who is credited with coining the term artificial intelligence in the proposal for the Dartmouth summer research project?

A. Geoffrey Hinton

B. Andrew Ng

C. John McCarthy

D. Jürgen Schmidhuber

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: John McCarthy

John McCarthy is credited with coining the term in the 1955 proposal for the Dartmouth summer research project. The workshop itself took place in 1956 and helped establish the research field. This naming attribution does not make McCarthy the sole inventor of every AI idea or method. The question now asks for the documented attribution rather than an oversimplified invention claim.

Why each choice is right or wrong

A - Incorrect: The historical naming attribution in the note is to McCarthy, not Hinton.

B - Incorrect: The historical naming attribution in the note is to McCarthy, not Ng.

C - Correct: McCarthy is credited with introducing the term in the 1955 Dartmouth proposal.

D - Incorrect: The historical naming attribution in the note is to McCarthy, not Schmidhuber.

Primary note proof

Study-note section AI-S04

> John McCarthy is credited with coining the term artificial intelligence in the 1955 proposal for the Dartmouth summer research project. The Dartmouth workshop took place in 1956 and helped establish the field. Naming the field does not mean that one person invented all its ideas or methods.

Study-note section AI43

> Face recognition is both an applied task and a narrow capability. Broad machine-intelligence goals can include emulating human intelligence and building systems that learn. Humanlike problem-solving alone does not uniquely select expert systems. Python is widely used, but C++ and Java also implement AI; “commonly used” does not isolate one language. McCarthy is associated with naming AI, not its sole invention.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 4 | lines 26-27

is concerned with the design of intelligence in an artificial device. The term was coined by McCarthy
in 1956. There are two ideas in the definition. i.e Intelligence, and Artificial device.

Question sources

Artificial Intelligence practice questions · Question 22

Question 72 | Q01029

Which activity, as stated, directly requires an intelligence-related capability, rather than merely naming a computing operation that may or may not use AI?

A. It helps to exploit vulnerabilities to secure the firm

B. Language understanding and problem-solving (Text analytics and NLP)

C. Easy to create a website

D. It helps to deploy applications on the cloud

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer B: Language understanding and problem-solving (Text analytics and NLP)

AI concerns capabilities such as reasoning, learning and language understanding. The language-understanding and problem-solving option explicitly names such capabilities. Website creation, cloud deployment and security operations can incorporate AI, but the activity names alone do not establish its use. The distinction is between an explicit intelligence-related task and a computing context that could host many kinds of software.

Why each choice is right or wrong

A - Incorrect: Security testing or exploitation can use ordinary programmed tools; the activity alone does not require intelligent learning or reasoning.

B - Correct: Understanding human language and solving problems directly name intelligence-related capabilities.

C - Incorrect: Creating a website can use conventional software tools without intelligent behavior.

D - Incorrect: Deploying an application to a cloud platform does not itself require an intelligent application.

Primary note proof

Study-note section AI01

Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI37

NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 6 | lines 18-19

• Understanding language involving natural language processing, speech processing
• Solving problems

Introduction to Artificial Intelligence module | PDF page 6 | lines 19-20

• Solving problems
• Robotics

Question sources

Artificial Intelligence practice questions · Question 23

Question 73 | Q01030

Which of the following is a component of Artificial Intelligence?

A. Learning

B. Training

C. Designing

D. Puzzling

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Learning

Learning appears explicitly among intelligent behaviors (PDF p. 6), and is defined as improving performance from experience on p. 67. The other words are not offered as parallel components in the module.

Why each choice is right or wrong

A - Correct: Explicitly named intelligent capability. See PDF p. 6.

B - Incorrect: Training is a model-building activity, not the named component here. See PDF p. 6.

C - Incorrect: Designing is not presented as the matching intelligent capability. See PDF p. 6.

D - Incorrect: Puzzling is not a defined component. See PDF p. 6.

Primary note proof

Study-note section AI08

> The performance element selects actions. The learning element changes the agent to improve future performance. The critic evaluates behavior against a performance standard and supplies feedback. The problem generator suggests exploratory actions likely to yield useful experience.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 6 | lines 17-19

> • Learning
> • Understanding language involving natural language processing, speech processing
> • Solving problems

Question sources

Artificial Intelligence practice questions · Question 24

Question 74 | Q01032

Which of the following is not a type of Artificial Intelligence agent?

A. Learning AI agent

B. Goal-based AI agent

C. Simple reflex AI agent

D. Unity-based AI agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Unity-based AI agent

The module lists simple reflex, model-based reflex, goal-based, utility-based and learning agents (PDF pp. 14-15). “Unity-based” is not “utility-based” and is not one of those classes.

Why each choice is right or wrong

A - Incorrect: Learning is a listed class. See PDF p. 14, 15.

B - Incorrect: Goal-based is listed. See PDF p. 14, 15.

C - Incorrect: Simple reflex is listed. See PDF p. 14, 15.

D - Correct: Unity-based is not a listed class. See PDF p. 14, 15.

Primary note proof

Study-note section AI42

> The common teaching list contains five agent classes when learning agents are included, or four decision architectures before learning is added. State the convention when counting agent types. “Target-based” can mean goal-based; “unity-based” is not utility-based.

Study-note section AI06

> A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean. It has no persistent world-state memory. This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

Study-note section AI08

> The performance element selects actions. The learning element changes the agent to improve future performance. The critic evaluates behavior against a performance standard and supplies feedback. The problem generator suggests exploratory actions likely to yield useful experience.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 14 | lines 10-11

> Agents can be grouped into five classes based on their degree of perceived intelligence and capability.
> All these agents can improve their performance and generate better action over the time.

Introduction to Artificial Intelligence module | PDF page 15 | lines 20-21

> 2.9.5 Learning agent
> • Learning allows an agent to operate in initially unknown environments.

Question sources

Artificial Intelligence practice questions · Question 26

Question 75 | Q01033

Which agent architecture applies fixed condition-action rules directly to the current percept, without retaining an internal world-state model?

A. Simple reflex agent

B. Model-based agent

C. Goal-based agent

D. Utility-based agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Simple reflex agent

The absence of persistent internal state is the key distinction. A simple reflex agent maps the current percept to an action using condition-action rules. Model-based reflex agents may also use rules, but they maintain a world-state representation. Goal-based and utility-based agents use goals or outcome preferences to guide selection, rather than being defined by current-percept-only rule matching.

Why each choice is right or wrong

A - Correct: A simple reflex agent chooses from the current percept without persistent world-state memory.

B - Incorrect: A model-based reflex agent adds internal state and applies rules to that modeled state.

C - Incorrect: Goal-based selection evaluates actions or future sequences against a desired state.

D - Incorrect: Utility-based selection ranks outcomes by desirability or expected utility.

Primary note proof

Study-note section AI06

> A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean. It has no persistent world-state memory. This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

Study-note section AI06

> A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects. It then applies rules to that state. Both kinds can use condition-action rules; memory and a world model distinguish them.

Study-note section AI07

> A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: $EU(a)=\Sigma s\ P(s|a,evidence)U(s)$. Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 14 | lines 17-18

> • The Simple reflex agent works on Condition-action rule, which means it maps the current state to
> action.

Introduction to Artificial Intelligence module | PDF page 14 | lines 29-30

- Internal State: It is a representation of the current state based on percept history.
• These agents have the model, "which is knowledge of the world" and based on the model they

Question sources

Artificial Intelligence practice questions · Question 27

Question 76 | Q01034

Which type of agent maintains an internal model of the world to make decisions?

A. Simple reflex agent

B. Model-based agent

C. Goal-based agent

D. Utility-based agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Model-based agent

An internal state based on percept history and a model of how the world changes define the model-based category. Goal and utility agents can extend that architecture, so the requested defining type is B. Evidence: PDF 14.

Why each choice is right or wrong

A - Incorrect: Current-percept rules omit history. See PDF p. 14.

B - Correct: Matches the internal model definition. See PDF p. 14.

C - Incorrect: Adds explicit goal information to a model. See PDF p. 14.

D - Incorrect: Adds a numerical utility criterion. See PDF p. 14.

Primary note proof

Study-note section AI06

A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects. It then applies rules to that state. Both kinds can use condition-action rules; memory and a world model distinguish them.

Study-note section AI07

A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: $EU(a)=\Sigma s\ P(s|a,evidence)U(s)$. Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 14 | lines 29-31

- Internal State: It is a representation of the current state based on percept history.
• These agents have the model, "which is knowledge of the world" and based on the model they perform actions.

Question sources

Artificial Intelligence practice questions · Question 28

Question 77 | Q01035

Which type of agent selects actions based on their expected utility or desirability?

A. Simple reflex agent

B. Model-based agent

C. Goal-based agent

D. Utility-based agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Utility-based agent

A utility function assigns real numbers to states to compare how efficiently actions achieve goals; this is precisely desirability-based selection. Evidence: PDF 15.

Why each choice is right or wrong

A - Incorrect: Uses condition-action rules. See PDF p. 15.

B - Incorrect: A world model alone supplies no preference scale. See PDF p. 15.

C - Incorrect: Goal achievement alone does not rank desirable outcomes. See PDF p. 15.

D - Correct: Numerical utility ranks alternatives. See PDF p. 15.

Primary note proof

Study-note section AI07

> A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: EU(a)=Σs P(s|a,evidence)U(s). Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 15 | lines 18-20

> • The utility function maps each state to a real number to check how efficiently each action achieves
> the goals.
> 2.9.5 Learning agent

Question sources

Artificial Intelligence practice questions · Question 29

Question 78 | Q01036

Which type of agent represents goals and selects actions that maximize the achievement of those goals?

A. Simple reflex agent

B. Model-based agent

C. Goal-based agent

D. Utility-based agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer C: Goal-based agent

Explicit desirable goal situations and choosing actions to reach them characterize goal-based agents. Evidence: PDF 15.

Why each choice is right or wrong

A - Incorrect: Responds to current percept. See PDF p. 15.

B - Incorrect: Tracks state without necessarily representing a goal. See PDF p. 15.

C - Correct: Explicit goal information supplies the requested criterion. See PDF p. 15.

D - Incorrect: Adds utility rankings beyond mere goal achievement. See PDF p. 15.

Primary note proof

Study-note section AI07

> A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: EU(a)=Σs P(s|a,evidence)U(s). Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Study-note section AI07

> Updating an internal state after selecting an action predicts the action’s effect; it does not itself modify the external environment. A stored variable named goal is insufficient to prove a goal-based policy if BEST_ACTION never uses it. Model-based, goal-based, utility-based and learning capabilities can coexist.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 15 | lines 6-8

> • Goal-based agents expand the capabilities of the model-based agent by having the "goal" information.
> • They choose an action, so that they can achieve the goal.

Question sources

Artificial Intelligence practice questions · Question 30

Question 79 | Q01037

Which type of agent improves its performance over time through learning from experience?

A. Simple reflex agent

B. Model-based agent

C. Goal-based agent

D. Learning agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Learning agent

Learning agents adapt using past experience and a learning element that changes performance. Evidence: PDF 15.

Why each choice is right or wrong

A - Incorrect: Current rules do not themselves explain improvement. See PDF p. 15.

B - Incorrect: Tracking state is not itself learning. See PDF p. 15.

C - Incorrect: Goal pursuit need not change performance over time. See PDF p. 15.

D - Correct: Explicitly learns from experience. See PDF p. 15.

Primary note proof

Study-note section AI08

> The performance element selects actions. The learning element changes the agent to improve future performance. The critic evaluates behavior against a performance standard and supplies feedback. The problem generator suggests exploratory actions likely to yield useful experience.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 15 | lines 24-26

• A learning agent in AI is the type of agent which can learn from its past experiences, or it has learning
capabilities.
• It starts to act with basic knowledge and then able to act and adapt automatically through learning.

Question sources

Artificial Intelligence practice questions · Question 31

Question 80 | Q01038

A face-recognition system is built to control entry to a building, without trying to explain biological recognition processes. Which approach classifies this practical engineering purpose, rather than its breadth of capability?

A. Weak AI approach

B. Applied AI approach

C. Cognitive AI approach

D. Strong AI approach

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: Applied AI approach

Applied AI builds useful solutions to practical tasks, so building-access recognition fits that purpose. The same system is also narrow or weak AI in capability scope. These labels answer different classification questions and are compatible. Cognitive AI would investigate biological intelligence, while strong-AI claims concern broader competence or actual understanding. The explicit purpose axis isolates the intended selection.

Why each choice is right or wrong

A - Incorrect: Weak AI describes bounded capability, which also fits face recognition, but does not classify the purpose requested.

B - Correct: Applied AI builds a useful practical solution, matching the entry-control purpose.

C - Incorrect: Cognitive AI investigates biological intelligence; that is not the stated purpose.

D - Incorrect: Strong AI concerns actual understanding or broad intelligence claims, not the application purpose described.

Primary note proof

Study-note section AI01

Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses

computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI01

Narrow or weak AI addresses bounded tasks, such as face recognition, spam filtering or a voice assistant. General AI denotes broad competence transferable across domains; strong AI also has a philosophical meaning involving actual understanding. Do not equate a humanoid body, autonomous vehicle, or many features with general intelligence.

Study-note section AI43

Face recognition is both an applied task and a narrow capability. Broad machine-intelligence goals can include emulating human intelligence and building systems that learn. Humanlike problem-solving alone does not uniquely select expert systems. Python is widely used, but C++ and Java also implement AI; “commonly used” does not isolate one language. McCarthy is associated with naming AI, not its sole invention.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 7 | lines 6-8

Applied AI: aims to produce commercially viable smart system. For example, a security system that
is able to recognize the faces of people who are permitted to enter a particular building. Applied AI
has already enjoyed considerable success.

Question sources

Artificial Intelligence practice questions · Question 32

Question 81 | Q01039

Which single option gives the complete set of possible AI benefits in suitable, evaluated applications? Interpret the choices operationally, without assuming consciousness or guaranteed improvement.

A. Can reduce search work and time with an effective heuristic.

B. Can support protective filtering by detecting unwanted spam messages.

C. Can automate reasoning and learning to help solve tasks.

D. All of the above

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer D: All of the above

Useful AI methods can reduce search work when a heuristic guides exploration effectively. Learned detection of unwanted spam can support protective filtering, and computational reasoning or learning can automate parts of a task. These are possible operational benefits rather than guarantees for every deployment or claims of consciousness. Each individual choice gives only one benefit, so the inclusive response supplies the complete set.

Why each choice is right or wrong

A - Incorrect: Effective heuristics can reduce explored work, but this is only one of the listed possible benefits.

B - Incorrect: Spam classification can support protective filtering, but this alone is incomplete.

C - Incorrect: Automating computational reasoning and learning can help tasks, but this alone is incomplete.

D - Correct: All three are possible operational benefits under the stated suitable-application scope.

Primary note proof

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 7 | lines 6-8

> Applied AI: aims to produce commercially viable smart system. For example, a security system that
> is able to recognize the faces of people who are permitted to enter a particular building. Applied AI
> has already enjoyed considerable success.

Introduction to Artificial Intelligence module | PDF page 79 | lines 11-12

- It is very time efficient.
- Most of the companies use NLP to improve the efficiency of documentation processes,

Introduction to Artificial Intelligence module | PDF page 7 | lines 3-5

Weak AI: deals with the creation of some form of computer-based artificial intelligence that cannot
truly reason and solve problems, but can act as if it were intelligent. Weak AI holds that suitably programmed machines can simulate human cognition.

Question sources

Artificial Intelligence practice questions · Question 33

Question 82 | Q01040

Which of the following is/are the composition for AI agents?

A. Program only

B. Architecture only

C. Both Program and Architecture

D. None of the mentioned

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer C: Both Program and Architecture

The structural equation adds architecture and agent program; neither component alone is the complete agent. Evidence: PDF 12.

Why each choice is right or wrong

A - Incorrect: Omits executing machinery. See PDF p. 12.

B - Incorrect: Omits implementing program. See PDF p. 12.

C - Correct: Contains both terms in the equation. See PDF p. 12.

D - Incorrect: Contradicts the explicit equation. See PDF p. 12.

Primary note proof

Study-note section AI03

A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer’s prior instructions.

Study-note section AI-S03

> An AI task is modeled as an agent interacting with its environment: observations enter through sensors and selected actions affect the environment through actuators. This task-level description differs from an agent implementation, which combines a program with a computing architecture.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 12 | lines 25-27

> • Agent = Architecture + Agent Program
> • Architecture = the machinery that an agent executes on.
> • Agent Program = an implementation of an agent function.

Question sources

Artificial Intelligence practice questions · Question 34

Question 83 | Q01041

Which are examples of artificial agents, assuming the spacecraft and robot use constructed controllers that sense and select actions autonomously? Select all that apply.

A. Autonomous Spacecraft

B. Human

C. Robot

D. All of the mentioned

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A, C: Autonomous Spacecraft; Robot

An artificial agent uses a constructed sensing and action-selection system to interact with its environment. Under the explicit control assumption, both the autonomous spacecraft and robot qualify. Humans also perceive and act, so they are agents, but they are not artificial agents. The inclusive choice fails because it includes humans. This classification concerns artificial agency and does not assert that a robot has general intelligence.

Why each choice is right or wrong

A - Correct: An autonomous spacecraft uses constructed sensing and action-selection mechanisms to interact with its environment.

B - Incorrect: A human is an agent but is not an artificial agent.

C - Correct: An AI-controlled robot senses and acts physically and is an artificial-agent example.

D - Incorrect: Including humans makes this aggregate false under the question's artificial-agent wording.

Primary note proof

Study-note section AI03

> An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 12 | lines 2-4

> • Human: sensor (eye, ears, skins, taste). effectors (Hand, fingers, legs, mouth)
> • Robot: sensor (camera, infrared,..), effectors (gripper, wheel, speaker,.)
> • Software agent: function as sensor and actuator

Introduction to Artificial Intelligence module | PDF page 82 | lines 11-13

> efficient robots. Robots are the artificial agents acting in real world environment. Robots are aimed at
> manipulating the objects by perceiving, picking, moving, modifying the physical properties of object,
> destroying it, or to have an effect thereby freeing manpower from doing repetitive functions without

Question sources

Artificial Intelligence practice questions · Question 35

Question 84 | Q01042

Which of the following is an expansion of Artificial Intelligence application?

A. Game Playing

B. Planning and Scheduling

C. Diagnosis

D. All of the mentioned

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer D: All of the mentioned

The application list includes medical diagnosis, telescope planning/scheduling, and grand-master game play. All three categories are present. Evidence: PDF 7, 7.

Why each choice is right or wrong

A - Incorrect: Games are explicitly listed. See PDF p. 7, 7.

B - Incorrect: Planning and scheduling are explicitly listed. See PDF p. 7, 7.

C - Incorrect: Medical diagnosis is explicitly listed. See PDF p. 7, 7.

D - Correct: Combines all three documented applications. See PDF p. 7, 7.

Primary note proof

Study-note section AI-S03

> Planning constructs actions to reach a goal. Scheduling additionally assigns activities to times and resources while respecting constraints such as resource conflicts and precedence. It is an AI application when search or constraint reasoning chooses feasible or useful assignments; automated telescope experiment scheduling is an example.

Study-note section AI22

> Adversarial search models agents with competing objectives. A game specifies states, player to move, legal actions, successor function, terminal test and utility. Perfect information means previous moves and current state are visible; it does not reveal future choices or imply absence of chance. Zero-sum utility makes one player’s gain the other’s loss.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 7 | lines 22-24

> • Today‘s Expert systems can carry out medical diagnosis in a narrow domain
> • Speech understanding systems are capable of recognizing several thousand words continuous speech

Introduction to Artificial Intelligence module | PDF page 7 | lines 25-29

> • Planning and scheduling systems had been employed in scheduling experiments with the Hubble Telescope

• The Learning systems are capable of doing text categorization into about a 1000 topics
• In Games, AI systems can play at the Grand Master level in chess (world champion), checkers, etc.

Question sources

Artificial Intelligence practice questions · Question 36

Question 85 | Q01043

Which option gives the complete set of valid general descriptions or examples of artificial agents?

A. Receives input from its surroundings and selects operations that act on those surroundings.

B. A line-following robot controller that reads sensors and commands the robot's motors.

C. Perceives its environment through sensors and acting upon that environment through actuators

D. All of the mentioned

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: All of the mentioned

An agent perceives its environment and acts on it. The input-and-operation paraphrase describes that interaction, the sensor-driven line-following controller is an implementation example, and the sensor/actuator statement supplies the general definition. All are valid under the stated descriptions. Since the question requests the complete set in one selection, the aggregate is selected; each individual statement is true but incomplete.

Why each choice is right or wrong

A - Incorrect: This is a valid input-and-action paraphrase, but it alone omits the other valid descriptions.

B - Incorrect: The controller senses and acts through the robot, so it is a valid example, but alone is not the complete set.

C - Incorrect: This gives the general sensing-and-acting definition, but alone omits the valid paraphrase and example.

D - Correct: The paraphrase, robot-controller example and sensor/actuator definition all fit artificial agency.

Primary note proof

Study-note section AI03

An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Study-note section AI03

A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer's prior instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 10 | lines 21-23

• An agent is anything that can perceive its environment through sensors and acts upon that environment through effectors. An agent perceives its environment through sensors.
• The complete set of inputs at a given time is called a percept.

Question sources

Artificial Intelligence practice questions · Question 37

Question 86 | Q01044

What is the name of Artificial Intelligence which allows machines to handle vague information with a deftness that mimics human intuition?

A. Human intelligence

B. Boolean logic

C. Functional logic

D. Fuzzy logic

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer D: Fuzzy logic.

AI30 represents vague categories with degrees between zero and one and explains fuzzy control using graded rules. This supplies the concrete mechanism behind the stem's loose phrase about human intuition. Boolean logic uses only true or false. Human intelligence is not the named machine-reasoning formalism here, and functional logic is not the provided graded-membership method.

Why each choice is right or wrong

A - Incorrect: Human intelligence names biological capability, not the specified artificial graded-membership technique.

B - Incorrect: Boolean logic uses crisp true/false values rather than degrees of membership in vague categories.

C - Incorrect: Functional logic is not the graded-membership and fuzzy-rule formalism described in AI30.

D - Correct: Fuzzy logic handles vague linguistic categories through graded membership and graded rules.

Primary note proof

Study-note section AI30

> Fuzzy sets assign graded membership μ(x) between 0 and 1 to concepts such as warm or tall. A temperature can belong to warm with degree 0.7. This is not a 70% probability that an otherwise crisp warm proposition is true. Boolean logic uses only true and false; probability describes uncertainty about events.

Study-note section AI25

> Propositional atoms represent whole true-or-false claims. ¬P negates; P∧Q requires both; P∨Q requires at least one; P→Q is false only when P is true and Q false; P↔Q requires matching truth values. Thus P→Q ≡ ¬P∨Q. De Morgan gives ¬(P∧Q) ≡ ¬P∨¬Q.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 45 | lines 19-21

> • Propositional/Boolean logic as it works on 0 and 1.
> • In PL, we use symbolic variables to represent the logic, and we can use any symbol for a representing
> a proposition, such A, B, C, P, Q, R, etc.

Question sources

Artificial Intelligence practice questions · Question 38

Question 87 | Q01045

An AI system is composed of?

A. agent

B. environment

C. Both A and B

D. None of the Above

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer C: Both A and B

The explicit system composition is agent plus environment. Evidence: PDF 11.

Why each choice is right or wrong

A - Incorrect: Only one component. See PDF p. 11.

B - Incorrect: Only the other component. See PDF p. 11.

C - Correct: Exactly both documented components. See PDF p. 11.

D - Incorrect: Conflicts with the explicit composition. See PDF p. 11.

Primary note proof

Study-note section AI-S03

> An AI task is modeled as an agent interacting with its environment: observations enter through sensors and selected actions affect the environment through actuators. This task-level description differs from an agent implementation, which combines a program with a computing architecture.

Study-note section AI03

> A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer's prior instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 11 | lines 21-23

> • An AI system is composed of an agent and its environment.
> • The agents act in their environment. The environment may contain other agents.
> • An agent is anything that can perceive its environment through sensors and acts upon that

Question sources

Artificial Intelligence practice questions · Question 39

Question 88 | Q01046

What are Agent?

A. An agent perceives its environment through sensors and acts on it through actuators.

B. An agent is anything that can change its environment through sensors

C. An agent is anything that can control its environment through sensors

D. None of the Above

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: An agent perceives its environment through sensors and acts on it through actuators.

An agent is an entity that perceives an environment through sensors and acts on it through actuators or effectors. Sensing alone is only part of this definition, so the first option now includes acting. The other descriptions incorrectly give sensors the action or control role. An agent need not control everything in its environment or guarantee a desired outcome.

Why each choice is right or wrong

A - Correct: This states both parts of the agent interaction: sensing and acting.

B - Incorrect: Sensors supply observations; they are not the general components that execute environmental changes.

C - Incorrect: Sensors do not generally control the environment; action uses actuators or effectors.

D - Incorrect: A complete agent definition is supplied by the sensing-and-acting statement.

Primary note proof

Study-note section AI03

> An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 10 | lines 21-23

> • An agent is anything that can perceive its environment through sensors and acts upon that environment through effectors. An agent perceives its environment through sensors.
> • The complete set of inputs at a given time is called a percept.

Question sources

Artificial Intelligence practice questions · Question 40

Question 89 | Q01047

Which instruments are used for perceiving and acting upon the environment?

A. Sensors and Actuators

B. Sensors

C. Perceiver

D. Perceiver and Sensor

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Sensors and Actuators

Sensors receive environmental input and actuators deliver output, so both are needed. Evidence: PDF 13.

Why each choice is right or wrong

A - Correct: Includes perception and action interfaces. See PDF p. 13.

B - Incorrect: Omits action interface. See PDF p. 13.

C - Incorrect: Does not identify the two instruments. See PDF p. 13.

D - Incorrect: Two perception terms omit actuators. See PDF p. 13.

Primary note proof

Study-note section AI03

> An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Study-note section AI05

> PEAS means Performance measure, Environment, Actuators, Sensors. Specify all four before designing an agent. For a taxi: performance includes safety and travel time; environment includes roads, traffic and passengers; actuators include steering, brakes and displays; sensors include cameras, positioning and speed readings.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 13 | lines 6-8

> • The agent takes input from the environment through sensors and delivers the output to the environment through actuators.
> • The most famous artificial environment is the Turing Test environment, in which one real and

Question sources

Artificial Intelligence practice questions · Question 41

Question 90 | Q01048

In the teaching taxonomy of decision architectures before learning agents are added as a separate class, how many agent types are counted?

A. 1

B. 2

C. 3

D. 4

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: 4

The note distinguishes a four-architecture convention from a five-class list that also includes learning agents. The specified convention counts simple reflex, model-based reflex, goal-based and utility-based architectures. Learning can improve several of those architectures, but it is excluded as a separate class for this count. The explicit taxonomy therefore determines the number without claiming a universal count for every possible classification.

Why each choice is right or wrong

A - Incorrect: One does not match either teaching convention described in AI42.

B - Incorrect: Two does not match the stated architecture or learning-inclusive list.

C - Incorrect: Three is not either count given here.

D - Correct: The stated pre-learning architecture taxonomy contains four types.

Primary note proof

Study-note section AI42

> The common teaching list contains five agent classes when learning agents are included, or four decision architectures before learning is added. State the convention when counting agent types. “Target-based” can mean goal-based; “unity-based” is not utility-based.

Study-note section AI06

> A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean. It has no persistent world-state memory. This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

Study-note section AI07

A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: EU(a)=Σs P(s | a,evidence)U(s). Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 14 | lines 10-12

Agents can be grouped into five classes based on their degree of perceived intelligence and capability.
All these agents can improve their performance and generate better action over the time.
2.9.1 Simple reflex agent

Question sources

Artificial Intelligence practice questions · Question 42

Question 91 | Q01049

Which of the following is not an agent decision architecture?

A. Model based

B. Utility based

C. Simple reflex

D. Search frontier

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer D: Search frontier

Model-based, utility-based and simple reflex describe ways to organize agent decision making. A search frontier instead stores generated nodes awaiting selection or expansion during a search. It is an algorithm component rather than a decision architecture. The overlapping target-based distractor has been replaced because target-based can reasonably mean goal-based, which is a valid agent category.

Why each choice is right or wrong

A - Incorrect: A model-based architecture maintains an internal representation used in action selection.

B - Incorrect: A utility-based architecture selects using outcome desirability.

C - Incorrect: Simple reflex is a current-percept condition-action architecture.

D - Correct: A search frontier is the collection of generated candidates awaiting selection, not an agent decision architecture.

Primary note proof

Study-note section AI42

The common teaching list contains five agent classes when learning agents are included, or four decision architectures before learning is added. State the convention when counting agent types. “Target-based” can mean goal-based; “unity-based” is not utility-based.

Study-note section AI12

A search node records a state plus parent, action, depth and path cost. Expanding a node generates its successors. OPEN or frontier stores discovered candidates awaiting expansion; CLOSED records expanded states. Reaching the same state through different paths does not imply the same node or cost.

Study-note section AI06

A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean.

Study-note section AI06

A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects.

Study-note section AI07

A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: $EU(a)=\Sigma s\ P(s|a,evidence)U(s)$. Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 14 | lines 9-11

2.9 Agent Types
Agents can be grouped into five classes based on their degree of perceived intelligence and capability.
All these agents can improve their performance and generate better action over the time.

Introduction to Artificial Intelligence module | PDF page 15 | lines 2-3

2.9.3 Goal-based agent
• The knowledge of the current state environment is not always sufficient to decide for an agent to

Introduction to Artificial Intelligence module | PDF page 15 | lines 12-13

2.9.4 Utility-based agent
• These agents are similar to the goal-based agent but provide an extra component of utility

Question sources

Artificial Intelligence practice questions · Question 43

Question 92 | Q01050

Which is used to improve the agents performance?

A. Perceiving

B. Observing

C. Learning

D. Sequence

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer C: Learning

The learning element makes improvements from the environment; perception and observation supply data but are not the improvement mechanism. Evidence: PDF 15.

Why each choice is right or wrong

A - Incorrect: Supplies input without necessarily changing policy. See PDF p. 15.

B - Incorrect: Observation alone need not improve performance. See PDF p. 15.

C - Correct: Named improvement mechanism. See PDF p. 15.

D - Incorrect: A history is data, not the learning operation. See PDF p. 15.

Primary note proof

Study-note section AI08

> The performance element selects actions. The learning element changes the agent to improve future performance. The critic evaluates behavior against a performance standard and supplies feedback. The problem generator suggests exploratory actions likely to yield useful experience.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 15 | lines 28-30

> - Learning element is for making improvements by learning from environment
> - Critic Learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.

Question sources

Artificial Intelligence practice questions · Question 44

Question 93 | Q01051

Rationality is?

A. being reasonable

B. being sensible

C. having good sense of judgment.

D. All of the above

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: All of the above

The module defines rationality with all three phrases: reasonable, sensible, and good judgment. Evidence: PDF 11.

Why each choice is right or wrong

A - Incorrect: Part of the definition. See PDF p. 11.

B - Incorrect: Part of the definition. See PDF p. 11.

C - Incorrect: Part of the definition. See PDF p. 11.

D - Correct: Includes every listed phrase. See PDF p. 11.

Primary note proof

Study-note section AI-S03

> In ordinary usage, rationality means reasonable, sensible judgment. For an AI agent, the operational criterion is the action with the best expected performance given its percept history and available knowledge, rather than guaranteed success or knowledge of the actual future.

Study-note section AI04

> A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 11 | lines 27-29

• Rationalityis nothing but status of being reasonable, sensible, and having good sense of judgment.
• Rationality is concerned with expected actions and results depending upon what the agent has perceived.

Question sources

Artificial Intelligence practice questions · Question 45

Question 94 | Q01052

Rationality of an agent does not depends on?

A. performance measures

B. Percept Sequence

C. reaction

D. actions

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer C: reaction

The explicit determinants are performance measures, percept history, environmental knowledge, and available actions. Reaction is not a separate determinant in that list. Evidence: PDF 12, 12.

Why each choice is right or wrong

A - Incorrect: Explicit determinant. See PDF p. 12, 12.

B - Incorrect: Explicit determinant. See PDF p. 12, 12.

C - Correct: Not a separate listed factor; may colloquially refer to action. See PDF p. 12, 12.

D - Incorrect: Available actions explicitly matter. See PDF p. 12, 12.

Primary note proof

Study-note section AI04

A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 12 | lines 10-12

• Rationality of an agent depends on the following –
o The performance measures, which determine the degree of success.
o Agent‘s Percept Sequence till now.

Introduction to Artificial Intelligence module | PDF page 12 | lines 11-14

o The performance measures, which determine the degree of success.
o Agent‘s Percept Sequence till now.
o The agent‘s posterior knowledge about the environment.
o The actions that the agent can carry out.

Question sources

Artificial Intelligence practice questions · Question 46

Question 95 | Q01053

What is the full form of PEAS?

A. Perceptual Measure, Environment, Actuators, and Sensors

B. Performance Measure, Environment, Actuators, and Sensors

C. Performance Measure, Entity, Actuators, and Sensors

D. Performance Measure, Environment, Agent Function, and Sensors

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Performance Measure, Environment, Actuators, and Sensors

PEAS comprises Performance measure, Environment, Actuators, Sensors, spanning the page boundary. Evidence: PDF 13, 14.

Why each choice is right or wrong

A - Incorrect: Perceptual replaces Performance incorrectly. See PDF p. 13, 14.

B - Correct: All four components match. See PDF p. 13, 14.

C - Incorrect: Entity replaces Environment. See PDF p. 13, 14.

D - Incorrect: Agent Function replaces Actuators. See PDF p. 13, 14.

Primary note proof

Study-note section AI05

PEAS means Performance measure, Environment, Actuators, Sensors. Specify all four before designing an agent. For a taxi: performance includes safety and travel time; environment includes roads, traffic and passengers; actuators include steering, brakes and displays; sensors include cameras, positioning and speed readings.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 13 | lines 30-32

> 2.8 PEAS examples
> • The are four elements to take account when designing an agent to solve a particular problem
> • PEAS

Introduction to Artificial Intelligence module | PDF page 14 | lines 2-3

> o Actuator: agent acts with in its environment
> o Sensor: the agent senses its environment

Question sources

Artificial Intelligence practice questions · Question 47

Question 96 | Q01054

Chess is example of which properties?

A. Discrete

B. Continuous

C. Episodic

D. Non-deterministic

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Discrete

Chess is the explicit discrete-environment example: finitely many distinct states/actions. Evidence: PDF 13, 13.

Why each choice is right or wrong

A - Correct: Explicit chess classification. See PDF p. 13, 13.

B - Incorrect: Driving, not chess, is the continuous example. See PDF p. 13, 13.

C - Incorrect: Moves affect future positions rather than independent episodes. See PDF p. 13, 13.

D - Incorrect: Board transitions are determined by state and selected move. See PDF p. 13, 13.

Primary note proof

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities

vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 13 | lines 11-13

✓ Discrete / Continuous − If there are a limited number of distinct, clearly defined, states of the environment, the environment is discrete (For example, chess); otherwise it is continuous (For example, driving).

Introduction to Artificial Intelligence module | PDF page 13 | lines 23-28

✓ Deterministic / Non-deterministic − If the next state of the environment is completely determined
by the current state and the actions of the agent, then the environment is deterministic; otherwise it
is non-deterministic.
✓ Episodic / Non-episodic − In an episodic environment, each episode consists of the agent perceiving
and then acting. The quality of its action depends just on the episode itself. Subsequent episodes do
not depend on the actions in the previous episodes. Episodic environments are much simpler because

Question sources

Artificial Intelligence practice questions · Question 48

Question 97 | Q01055

For an untimed crossword puzzle whose clues and grid do not change while the solver deliberates, which environment property follows from that lack of change?

A. Dynamic

B. Static

C. Semi Dynamic

D. Continuous

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Static

A static environment remains unchanged while the agent deliberates. The untimed, fixed-grid puzzle meets that condition. A running clock that affects performance could make an otherwise fixed task semidynamic, but the stem excludes it. Dynamic change and continuous-valued state or action descriptions are separate properties and do not follow from the stated puzzle setting.

Why each choice is right or wrong

A - Incorrect: Dynamic means the environment can change during deliberation, excluded by the premise.

B - Correct: Static means the relevant environment does not change while the agent deliberates.

C - Incorrect: Semidynamic describes changing performance, such as a running clock, despite an otherwise fixed environment; none is present.

D - Incorrect: Continuity concerns real-valued states or actions, not whether the puzzle changes during deliberation.

Primary note proof

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 13 | lines 17-19

> ✓ Static / Dynamic – If the environment does not change while an agent is acting, then it is static; otherwise it is dynamic.
> ✓ Single agent / Multiple agents – The environment may contain other agents which may be of the

Question sources

Artificial Intelligence practice questions · Question 49

Question 98 | Q01056

Which learning regime improves an agent’s action choices through interaction with an environment that supplies rewards or penalties?

A. Supervised learning

B. Reinforcement learning

C. Semi-supervised learning

D. Unsupervised learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Reinforcement learning

Interaction alone is too broad to identify a learning regime. Here the environment supplies rewards or penalties for the agent's actions, and the agent improves its choices from that feedback. This defines reinforcement learning. Supervised and semi-supervised methods depend on target labels; unsupervised methods discover unlabeled structure. None of those label-based descriptions specifies this action-and-reward loop.

Why each choice is right or wrong

A - Incorrect: Supervised learning supplies correct target outputs rather than defining action-reward interaction.

B - Correct: Reinforcement learning improves decisions through environmental reward feedback.

C - Incorrect: Semi-supervised learning mixes labeled and unlabeled examples, not a defining action-reward objective.

D - Incorrect: Unsupervised learning seeks unlabeled structure without the stated reward-feedback objective.

Primary note proof

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI32

> Semi-supervised learning combines some labels with additional unlabeled data. Active learning chooses informative cases for an oracle to label; it still uses labels. "Exploratory learning" is not an exclusive standard synonym: it can mean unsupervised exploration or active data acquisition, so define the feedback. A task named summarization or time-series analysis is not automatically supervised; its training targets determine that.

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples.

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 15-17

> ⇒ Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action.
> The agent learns automatically with these feedbacks and improves its performance. In reinforcement

Question sources

Artificial Intelligence practice questions · Question 50

Question 99 | Q01057

Which is reinforcement learning's immediate parent field in the AI → machine learning → reinforcement learning hierarchy?

A. Artifical Intelligence

B. Data Analytics

C. Machine Learning

D. None of the above

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: Machine Learning

Reinforcement learning is a branch of machine learning, and machine learning is within artificial intelligence. Artificial intelligence therefore also contains reinforcement learning, but it is the broader ancestor rather than the immediate parent requested. Data analytics is an application context rather than the stated parent field. The explicit hierarchy makes the nearer containing field the unique taxonomic selection.

Why each choice is right or wrong

A - Incorrect: AI contains RL indirectly through machine learning, but is not the immediate parent requested.

B - Incorrect: Data analytics is not the stated parent-field classification of RL in the note.

C - Correct: Machine learning is the immediate parent of reinforcement learning in the stated hierarchy.

D - Incorrect: The immediate parent is present among the choices.

Primary note proof

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI43

ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 68 | lines 16-18

6.2 Types of learning (machine learning)
• Machine learning enables a machine to automaticallylearn from data, improve performance from
experiences, and predict things without being explicitly programmed.

Introduction to Artificial Intelligence module | PDF page 67 | lines 18-20

important feature of —intelligent‖. Machine learning is the subfield of AI concerned with intelligent
systems that learn from experience and examples. It is the computational study of algorithms that
improve performance based on experience

Question sources

Artificial Intelligence practice questions · Question 51

Question 100 | Q01058

Which type of feedback does an agent in reinforcement learning receive?

A. Predictions

B. Labels

C. Clusters

D. Rewards or penalties

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer D: Rewards or penalties

Rewards and penalties are the stated reinforcement feedback; supervised targets and unsupervised grouping differ. Evidence: PDF 69.

Why each choice is right or wrong

A - Incorrect: A prediction is output, not the reward signal. See PDF p. 69.

B - Incorrect: Labels belong to supervised examples. See PDF p. 69.

C - Incorrect: Clusters are an unsupervised result. See PDF p. 69.

D - Correct: Matches the explicit feedback description. See PDF p. 69.

Primary note proof

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 16-18

> reward for each right action and gets a penalty for each wrong action.
> The agent learns automatically with these feedbacks and improves its performance. In reinforcement
> learning, the agent interacts with the environment and explores it. The goal of an agent is to get the

Question sources

Artificial Intelligence practice questions · Question 52

Question 101 | Q01059

What is the main task of a problem-solving agent?

A. Solve the given problem and reach to goal

B. To find out which sequence of action will get it to the goal state

C. Both A and B

D. None of the Above

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Both A and B

A solution is an action sequence leading from start to goal; this supports both general goal achievement and finding that sequence. Evidence: PDF 18.

Why each choice is right or wrong

A - Incorrect: Describes the overall objective. See PDF p. 18.

B - Incorrect: Gives the operational search task. See PDF p. 18.

C - Correct: Both are supported by the solution definition. See PDF p. 18.

D - Incorrect: Contradicts the defined objective. See PDF p. 18.

Primary note proof

Study-note section AI11

> First formulate the goal; then formulate the problem by specifying initial state, legal actions, transition model, goal test, and action costs. Search selects a sequence; execution applies it. A state abstracts away irrelevant world details.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 18 | lines 19-21

> Solution: It is an action sequence which leads from the start node to the goal node.
> Optimal Solution: If a solution has the lowest cost among all solutions.
> 3.5 Search Strategies

Question sources

Artificial Intelligence practice questions · Question 53

Question 102 | Q01060

For DFS on an implicit tree with branching factor b and maximum depth m, storing pending siblings, which expression gives the standard worst-case space bound?

A. b

B. b^2

C. b^b

D. O(bm)

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer D: O(bm)

Depth-first search retains the active branch and pending alternatives rather than an entire breadth frontier. With up to b pending entries per level over depth m, the standard stored-frontier space bound is O(bm). Path-only on-demand generation can use O(m), but that is a different implementation convention. The expression gives the depth-dependent storage bound rather than an exponential time bound.

Why each choice is right or wrong

A - Incorrect: b omits growth with search depth m in the general DFS storage bound.

B - Incorrect: b^2 is not the stated depth-dependent worst-case DFS storage expression.

C - Incorrect: b^b is neither of the note's DFS space bounds.

D - Correct: With pending siblings, at most a branching-factor-sized set per depth gives the standard O(bm) space bound.

Primary note proof

Study-note section AI14

> DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Study-note section AI14

> Implicit-tree worst-case time is O(b^m), and space is O(bm) when pending siblings are stored, or O(m) for path-only recursive generation. DFS is not generally optimal and can fail on infinite paths or cycles. On a finite graph, visited-state tracking makes traversal complete but needs O(V) storage. Do not confuse the O(b^m) worst-case time bound with the DFS space bounds above.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 19 | lines 20-22

• Space Complexity: DFS algorithm needs to store only single path from the root node, hence space
complexity of DFS is equivalent to the size of the fringe set, which is O(bm).
• Optimal: DFS search algorithm is non-optimal, as it may generate a large number of steps or high

Question sources

Artificial Intelligence practice questions · Question 54

Question 103 | Q01061

For BFS on an implicit tree with branching factor b>1 and shallowest goal depth d, using generation-time goal testing, which expression gives the standard worst-case generated-node order?

A. b

B. b^d

C. b^2

D. b^b

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: b^d

BFS generates levels in order. Under generation-time goal testing, the generated nodes through the shallowest goal depth d are bounded by the geometric sum 1+b+...+b^d, whose dominant term is b^d when b>1. Testing only after expanding deeper nodes can change the conventional bound by another factor of b, so the question states the goal-test convention.

Why each choice is right or wrong

A - Incorrect: b counts one level and does not account for arbitrary goal depth.

B - Correct: The sum 1+b+...+b^d has order b^d for b>1.

C - Incorrect: b^2 only describes a fixed depth-two case rather than general depth d.

D - Incorrect: b^b wrongly replaces solution depth d with the branching factor.

Primary note proof

Study-note section AI13

The tree contains 1+b+...+b^d nodes through depth d, giving O(b^d) under generation-time goal testing. Expanding depth-d nodes before testing can generate depth d+1 and yields O(b^(d+1)); specify the convention. Explicit finite graph traversal with visited-state tracking takes O(V+E) time and O(V) space.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 20 | lines 20-22

> Time Complexity: is obtained by the number of nodes traversed in BFS until the shallowest Node.
> T
> (b) =O(bd
>) where d= depth of shallowest solution and b is a node at every state.

Question sources

Artificial Intelligence practice questions · Question 55

Question 104 | Q01062

Which labels describe the recursive DFS stack’s removal order? Select all that apply.

A. LIFO

B. LILO

C. FIFO

D. FILO

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A, D: LIFO; FILO

Recursive depth-first search returns first from the most recently entered unfinished call. That is last-in-first-out stack order, also called first-in-last-out. The two labels are synonyms rather than distinct data structures. FIFO or LILO describes the opposite end-to-end order and does not characterize the DFS call stack. The multiple-selection instruction therefore retains both correct names for the same discipline.

Why each choice is right or wrong

A - Correct: LIFO is the stack order used by recursion and depth-first exploration.

B - Incorrect: LILO does not express the required last-in-first-out removal order.

C - Incorrect: FIFO is queue order, associated with breadth-first rather than depth-first selection.

D - Correct: FILO is another name for LIFO and therefore also describes the DFS stack.

Primary note proof

Study-note section AI14

> DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Study-note section AI13

BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 18 | lines 34-35

- DFS uses a stack data structure for its implementation.
- The process of the DFS algorithm is similar to the BFS algorithm.

Question sources

Artificial Intelligence practice questions · Question 56

Question 105 | Q01063

Which of the following search algorithm searches forward from initial state and backward from goal state till both meet to identify a common state?

A. Uniform Cost Search

B. Iterative Deepening Depth-First Search

C. Bidirectional Search

D. None of the Above

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Bidirectional Search

The algorithm starts one search at the initial state and another at the goal and stops at their intersection. Evidence: PDF 23.

Why each choice is right or wrong

A - Incorrect: Orders by path cost rather than two directions. See PDF p. 23.

B - Incorrect: Repeats depth-limited search from the start. See PDF p. 23.

C - Correct: Exactly the two-front definition. See PDF p. 23.

D - Incorrect: C explicitly fits. See PDF p. 23.

Primary note proof

Study-note section AI16

Bidirectional search grows from start and goal until frontiers connect. With suitable reverse actions, meeting tests and equal-cost BFS, time and space may fall to O(b^(d/2)). Weighted variants require a sound stopping bound; the first intersection need not be cheapest. IDA* deepens an f=g+h threshold rather than only a depth limit.

Study-note section AI15

UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Study-note section AI16

DLS runs DFS to depth l and reports cutoff separately from failure. It misses goals deeper than l. Iterative deepening runs DLS with limits 0,1,2,...; for finite b it finds the shallowest solution with O(b^d) time and O(bd) frontier space, optimal for equal step costs. Repeated shallow work is small compared with the deepest level.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 23 | lines 16-18

• Bidirectional search algorithm runs two simultaneous searches, one form initial state called as forward-search and other from goal node called as backward-search, to find the goal node.
• Bidirectional search replaces one single search graph with two small sub-graphs in which one starts

Question sources

Artificial Intelligence practice questions · Question 57

Question 106 | Q01064

When is breadth-first search is optimal?

A. When there is less number of nodes

B. When all step costs are equal

C. When all step costs are unequal

D. All of the above

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: When all step costs are equal

Equal positive step costs make total path cost proportional to depth; BFS first reaches a shallowest solution and hence a cheapest one. Evidence: PDF 20.

Why each choice is right or wrong

A - Incorrect: Node count alone says nothing about cost. See PDF p. 20.

B - Correct: Equal step costs ensure depth order is cost order. See PDF p. 20.

C - Incorrect: Unequal costs can make a deeper path cheaper. See PDF p. 20.

D - Incorrect: A and C are not sufficient conditions. See PDF p. 20.

Primary note proof

Study-note section AI13

> BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI13

> The tree contains 1+b+...+b^d nodes through depth d, giving O(b^d) under generation-time goal testing. Expanding depth-d nodes before testing can generate depth d+1 and yields O(b^(d+1)); specify the convention. Explicit finite graph traversal with visited-state tracking takes O(V+E) time and O(V) space.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 20 | lines 28-30

> Optimality: BFS is optimal if path cost is a non-decreasing function of the depth of the node.
> 3. Depth limited search
> A depth-limited search algorithm is similar to depth-first search with a predetermined limit.
> Depth-

Question sources

Artificial Intelligence practice questions · Question 58

Question 107 | Q01065

What is the heuristic function of greedy best-first search?

A. f(n) != h(n)

B. f(n) < h(n)

C. f(n) = h(n)

D. f(n) > h(n)

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: f(n) = h(n)

The operational rule removes the node with lowest h(n), so the greedy evaluation is f(n)=h(n). The printed f(n)=g(n) on this same page conflicts with its own algorithm and nearby h definition; the operational evidence supports C. Evidence: PDF 25.

Why each choice is right or wrong

A - Incorrect: Inequality would not express that rule. See PDF p. 25.

B - Incorrect: Strictly smaller is not the definition. See PDF p. 25.

C - Correct: Matches ranking exclusively by h. See PDF p. 25.

D - Incorrect: Strictly greater is not the definition. See PDF p. 25.

Primary note proof

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 25 | lines 21-23

> o Step 3: Remove the node n, from the OPEN list which has the lowest value of h(n), and places it in the CLOSED list.
> o Step 4: Expand the node n, and generate the successors of node n.

Question sources

Artificial Intelligence practice questions · Question 59

Question 108 | Q01066

Greedy best-first search selects which descriptions of the next node? Interpret “closest” as estimated remaining cost h, not actual distance. Select all that apply.

A. Shallowest

B. Deepest

C. The one closest to the goal node

D. Minimum heuristic cost

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C, D: The one closest to the goal node; Minimum heuristic cost

Greedy best-first search orders the frontier by h alone, choosing the smallest estimated remaining cost and ignoring cost already paid. Under the explicit interpretation, estimated closest to the goal and minimum heuristic cost express that same rule. Shallowest-first uses depth as in breadth-first search, while deepest-first follows depth-first ordering. Both heuristic descriptions are therefore valid and neither depth description is selected.

Why each choice is right or wrong

A - Incorrect: Shallowest-first is BFS-style depth ordering rather than greedy heuristic selection.

B - Incorrect: Deepest-first is DFS-style selection rather than minimum estimated remaining cost.

C - Correct: Read closest as estimated by h; it is the same greedy criterion as minimum heuristic cost.

D - Correct: Greedy best-first selects a node with minimum h, ignoring already-paid path cost.

Primary note proof

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI13

> BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 25 | lines 14-16

• In the best first search algorithm, we expand the node which is closest to the goal node and the closest
cost is estimated by heuristic function-
• f(n)= g(n). Were, h(n)= estimated cost from node n to the goal.

Introduction to Artificial Intelligence module | PDF page 25 | lines 21-22

o Step 3: Remove the node n, from the OPEN list which has the lowest value of h(n), and places it in the CLOSED list.

Question sources

Artificial Intelligence practice questions · Question 60

Question 109 | Q01067

Which general search family contains A* because it selects frontier nodes by a minimum evaluation value?

A. Breadth-First-Search

B. Depth-First-Search

C. Uniform Cost Search

D. Best-First-Search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer D: Best-First-Search

Best-first search is the general family that selects a frontier node according to an evaluation value. A* is one member, using f=g+h. Uniform-cost search is another member, using g alone. Breadth-first and depth-first names describe depth-oriented selection disciplines. The question asks for the containing family rather than claiming A* is identical to one simpler algorithm.

Why each choice is right or wrong

A - Incorrect: BFS selects by depth/FIFO order rather than the general evaluation-priority family.

B - Incorrect: DFS follows stack/deepest-branch order rather than minimum evaluation priority.

C - Incorrect: UCS is another best-first method using g alone; it is not the general family containing A*.

D - Correct: Best-first search is the family that selects by an evaluation, with A* using g+h.

Primary note proof

Study-note section AI18

A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI15

UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Study-note section AI17

Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI13

BFS removes the oldest frontier entry using a FIFO queue.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 26 | lines 25-27

• A* search is the most commonly known form of best-first search.
• It uses heuristic function h(n), and cost to reach the node n from the start state g(n). It has combined
features of UCS and greedy best-first search, by which it solves the problem efficiently.

Question sources

Artificial Intelligence practice questions · Question 61

Question 110 | Q01188

Function MY_AGENT (PERCEPT) return action
static state, a description of the current world state
goal, a description of the goal to achieve may be in terms of state
state ← UPDATE_STATE(state, percept)
actionSet ← POSSIBLE_ACTIONS(state)
action ← BEST_ACTION(actionSet)
state ← UPDATE_STATE(state, action)
return action
What is the purpose of the final instruction state ← UPDATE_STATE(state, action), after BEST_ACTION has selected an action?

A. Modify the state of the environment if necessary

B. Update the state information to be like the one after executing the best action chosen

C. Update the environment so that the agent can perceive the next percept

D. Modify the state of the environment once the decision of which action to choose is made.

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Update the state information to be like the one after executing the best action chosen

The final update uses the chosen action to predict what the agent's internal state representation should look like after that action. It is distinct from the earlier update using the current percept. Updating memory does not execute an actuator or directly modify the external environment. Naming the exact instruction resolves the missing referent while preserving the model-update interpretation of the original prototype.

Why each choice is right or wrong

A - Incorrect: Updating an internal state representation does not itself modify the external environment.

B - Correct: The final call predicts and records the internal state after the selected action.

C - Incorrect: Neither internal update executes an environmental action or produces the next percept.

D - Incorrect: Selecting an action and updating memory are distinct from changing the external world through execution.

Primary note proof

Study-note section AI07

Updating an internal state after selecting an action predicts the action's effect; it does not itself modify the external environment. A stored variable named goal is insufficient to prove a goal-based policy if BEST_ACTION never uses it. Model-based, goal-based, utility-based and learning capabilities can coexist.

Study-note section AI06

A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects. It then applies rules to that state. Both kinds can use condition-action rules; memory and a world model distinguish them.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 14 | lines 32-34

• Updating the agent state requires information about:
- How the world evolves
- How the agent's action affects the world.

Question sources

Computer Science model exam 2 · Question 9

Question 111 | Q01194

To implement greedy best-first search, let the evaluation be the estimated remaining cost h(n). Which OPEN-list insertion policy supplies its node-selection order?

A. Append newly generated nodes at the beginning of the open list

B. Append newly generated nodes at the end of the open list

C. Insert newly generated nodes in the open list according to their path cost (lowest values first)

D. Insert newly generated nodes in the open list according to their evaluation function (lowest values first)

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer D: Insert newly generated nodes in the open list according to their evaluation function (lowest values first)

Greedy best-first search orders its frontier by h alone, the estimate of remaining cost. A priority-ordered OPEN list with the smallest evaluation first therefore implements the rule when that evaluation is h. Ordering by g would instead implement uniform-cost search. Unconditionally prepending or appending nodes gives stack-like or FIFO behavior without the required heuristic ranking.

Why each choice is right or wrong

A - Incorrect: Prepending every new node gives a stack-like policy rather than ordering by remaining-cost estimates.

B - Incorrect: Appending every new node gives FIFO breadth order rather than heuristic priority.

C - Incorrect: Ordering by accumulated path cost g implements uniform-cost selection, not greedy h-only selection.

D - Correct: Ordering by the stated evaluation h puts the smallest estimated remaining cost first.

Primary note proof

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI15

> UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Study-note section AI13

> BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI14

> DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 25 | lines 21-23

> o Step 3: Remove the node n, from the OPEN list which has the lowest value of h(n), and places it in the CLOSED list.
> o Step 4: Expand the node n, and generate the successors of node n.

Question sources

Computer Science model exam 2 · Question 15

Question 112 | Q01199

In classical propositional logic over a nonempty set of interpretations, which statements are false? In D, “models” means all interpretations. Select all that apply.

A. A sentence is said to be invalid iff it is unsatisfiable.

B. A sentence is valid if it is satisfiable.

C. A sentence which is not invalid is satisfiable.

D. A sentence is unsatisfiable if it is false in all models.

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer A, B: A sentence is said to be invalid iff it is unsatisfiable.; A sentence is valid if it is satisfiable.

Validity requires truth in every interpretation, while satisfiability requires truth in at least one. A contingent atom can therefore be satisfiable and invalid, refuting both the proposed invalid-unsatisfiable equivalence and the claim that satisfiability implies validity. Not invalid means valid and hence satisfiable in the stated nonempty setting. Unsatisfiable means false in every interpretation, so the remaining two statements are true.

Why each choice is right or wrong

A - Correct: False: a satisfiable contingent sentence can be invalid without being unsatisfiable.

B - Correct: False: truth in one interpretation does not imply truth in every interpretation.

C - Incorrect: True: not invalid means valid, which entails satisfiability in the ordinary model setting.

D - Incorrect: True: unsatisfiable means there is no interpretation making the sentence true.

Primary note proof

Study-note section AI25

> A model is an interpretation making a sentence true. Valid means true in every interpretation; satisfiable means true in at least one; unsatisfiable means true in none. Invalid means not valid, and may still be satisfiable. P∨¬P is valid; P is satisfiable but not valid; P∧¬P is unsatisfiable. Entailment KB⊨α means every model of KB also satisfies α.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 45 | lines 27-29

> • A proposition formula which is always true is called tautology, and it is also called a valid sentence.
> • A proposition formula which is always false is called Contradiction.
> • A proposition formula which has both true and false values is called Contingency.

Question sources

Computer Science model exam 2 · Question 20

Question 113 | Q01202

Consider this agent prototype:
Function MY_AGENT(percept) returns action
static state, goal
state ← UPDATE_STATE(state, percept)
actionSet ← POSSIBLE_ACTIONS(state)
action ← BEST_ACTION(actionSet, state, goal)
state ← UPDATE_STATE(state, action)
return action
BEST_ACTION uses lookahead over future action sequences and chooses by whether the specified desired state is reached; it does not rank successful outcomes by utility or learn from experience. Which category best describes this action-selection criterion?

A. Goal based

B. Model based reflex

C. Simple reflex

D. Utility based

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Goal based

The action selector explicitly consults the desired state and looks ahead over sequences that can reach it. That is the defining goal-based decision criterion. Persistent state also gives the agent a model-based capability, but memory alone does not identify its selection policy. A reflex policy would apply condition-action rules instead of goal-directed lookahead, and utility-based choice would rank degrees of desirability among outcomes.

Why each choice is right or wrong

A - Correct: Goal-directed lookahead chooses sequences reaching a desired state, matching goal-based action selection.

B - Incorrect: The agent maintains a model, but the stated policy uses goal-directed lookahead rather than reflex rules.

C - Incorrect: Persistent state and future-sequence evaluation contradict a current-percept-only reflex policy.

D - Incorrect: No utility ranking is used; the selector evaluates attainment of the desired state.

Primary note proof

Study-note section AI07

A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: EU(a)=Σs P(s | a,evidence)U(s). Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Study-note section AI07

Updating an internal state after selecting an action predicts the action's effect; it does not itself modify the external environment. A stored variable named goal is insufficient to prove a goal-based policy if BEST_ACTION never uses it. Model-based, goal-based, utility-based and learning capabilities can coexist.

Study-note section AI06

A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean. It has no persistent world-state memory. This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 15 | lines 6-8

• Goal-based agents expand the capabilities of the model-based agent by having the "goal" information.
• They choose an action, so that they can achieve the goal.

Introduction to Artificial Intelligence module | PDF page 14 | lines 29-31

- Internal State: It is a representation of the current state based on percept history.
• These agents have the model, "which is knowledge of the world" and based on the model they perform actions.

Question sources

Computer Science model exam 2 · Question 23

Question 114 | Q01220

Which are general terms for the components through which an arbitrary AI agent acts on its environment, rather than sensing components or one particular robot subtype? Select all that apply.

A. Sensors

B. Effectors

C. Actuators

D. Manipulators

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B, C: Effectors; Actuators

Actuators and effectors are equivalent general names for the agent's acting interface here. Sensors instead supply observations. A manipulator can be a particular physical acting device, but it is not the general term covering every agent, including software agents. The scope allows the two synonymous general terms without incorrectly denying that robotic manipulators can carry out actions.

Why each choice is right or wrong

A - Incorrect: Sensors receive observations rather than enact the agent's selected environmental changes.

B - Correct: Effectors are an accepted name for the components through which the agent acts.

C - Correct: Actuators are the note's equivalent general term for acting components.

D - Incorrect: A manipulator is a particular physical device, not the general acting-interface term requested.

Primary note proof

Study-note section AI03

> An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Study-note section AI42

> A generic rule does not distinguish simple from model-based reflex without specifying memory. A goal declaration unused by the action policy does not establish goal-based behavior. "Sensors and effectors" describes interaction, not omniscience or intelligence by itself. Define the mechanical distinction when comparing effectors and actuators as acting components.

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 10 | lines 25-27

- The agent can change the environment through actuators or effectors.
- An operation involving an effector is called an action.
- Actions can be grouped into action sequences.

Question sources

Computer Science model exam 2 · Question 41

Question 115 | Q01252

A control agent has incomplete observations, uncertain transition outcomes, actions that affect later decisions, a world that changes during deliberation, and real-valued measurements and controls. Which list matches all these properties?

A. Fully observable, discrete, episodic, static.

B. Static, episodic, continuous, partially observable.

C. Partially observable, non-deterministic, non-episodic, dynamic, continuous.

D. Deterministic, sequential (non-episodic), dynamic, continuous.

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer C: Partially observable, non-deterministic, non-episodic, dynamic, continuous.

The scenario establishes each environment dimension directly. Incomplete observations give partial observability; uncertain outcomes give non-determinism; effects on later decisions give sequentiality; ongoing change gives dynamics; and real-valued quantities give continuity. Only one list matches the whole combination. This classification avoids claiming that these dimensions establish a universal numerical ranking of task difficulty.

Why each choice is right or wrong

A - Incorrect: Full observability, episodic independence and a static discrete setting contradict the scenario.

B - Incorrect: Partial observability and continuity fit, but static and episodic do not.

C - Correct: Partial observability, non-determinism, sequentiality, dynamics and continuity all match.

D - Incorrect: The other stated properties fit, but determinism contradicts uncertain transition outcomes.

Primary note proof

Study-note section AI09

Fully observable means sensors reveal relevant state; partially observable means relevant information is hidden or noisy. Deterministic means state and action determine the next state;

stochastic means outcomes involve randomness. Strategic uncertainty about an opponent is distinct from dice-like randomness.

Study-note section AI09

Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 16 | lines 17-19

✓ The most challenging environments are partially observable, stochastic, sequential, dynamic, and
continuous, and contain multiple intelligent agents.
2.11 Questions

Question sources

Computer Science model exam 2 · Question 73

Question 116 | Q01279

Which described training tasks are supervised learning? Select all that apply.

A. Summarization trained on document/reference-summary pairs.

B. Time-series forecasting trained on history/next-value target pairs.

C. Sequence discovery without target labels.

D. Clustering unlabeled examples.

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer A, B: Summarization trained on document/reference-summary pairs.; Time-series forecasting trained on history/next-value target pairs.

Supervised training uses paired inputs and target outputs. Document/reference-summary pairs therefore make the summarization task supervised, and history/next-value pairs make the forecasting task supervised. Sequence discovery and clustering without targets instead seek structure in unlabeled data. The labels in the descriptions determine the regime; task names alone would not establish whether summarization or time-series analysis is supervised.

Why each choice is right or wrong

A - Correct: The reference summary is a target output paired with each input document.

B - Correct: The next value is a numeric target paired with each input history.

C - Incorrect: Without target labels, sequence discovery seeks unlabeled structure.

D - Incorrect: Clustering unlabeled examples is unsupervised structure discovery.

Primary note proof

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 5-7

> o Classification
> o Regression
> ⇒ Unsupervised learning means we are only given the x‘s. In either case, the goal is to estimate f.

Question sources

Computer Science model exam 2 · Question 100

Question 117 | Q01485

A common name used to describe search-based optimization technique based on the principles of Genetics and Natural Selection is ____.

A. Genetic algorithms

B. Heuristic algorithm

C. Hill climbing

D. Simulated annealing

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: Genetic algorithms.

AI20 names the search method with population, fitness, parent selection, crossover and mutation as a genetic algorithm. That is the specific common name requested by the genetics/natural-selection description. Heuristic algorithm is a broad umbrella rather than this name; hill climbing follows improved neighbors, and simulated annealing is not the population-selection mechanism described.

Why each choice is right or wrong

A - Correct: This is the specific name of the evolutionary search technique based on fitness, selection and genetic variation.

B - Incorrect: Heuristic algorithm is a broad category and does not specifically name the genetics-based population method.

C - Incorrect: Hill climbing improves a candidate through neighbors rather than maintaining the described genetic population.

D - Incorrect: Simulated annealing is not the named selection/crossover/mutation method defined in the note.

Primary note proof

Study-note section AI20

> A genetic algorithm maintains a population, evaluates fitness, selects parents, applies crossover and mutation, and repeats replacement. It searches without guaranteeing global optimality. Means-ends analysis selects a difference between current and goal states, chooses an operator reducing it, and recursively solves its preconditions. Neither method is simply A* running backward.

Study-note section AI20

> Local search keeps a complete candidate configuration and improves its objective; the path may be irrelevant. Hill climbing chooses better neighbors. First-improvement accepts the first improvement; steepest-ascent examines all neighbors; stochastic variants choose among improvements. Local maxima, plateaus and ridges can stop progress. Random restarts or limited sideways steps can help without guaranteeing success.

Study-note section AI-S05

> Simulated annealing follows a current candidate and randomly samples a neighboring move. It accepts improvements and sometimes accepts worse moves according to a temperature that decreases over time. Unlike genetic search, its defining mechanism is not a population undergoing parent selection, crossover and mutation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 8 | lines 36-38

o Genetic Algorithms
1. Which one of the following is correctly matched with the definition and approaches of AI
A)Acting humanly - The rational agent approach

Question sources

Mock exam 1 · Question 1

Question 118 | Q01486

A complete history of everything that the agent has ever perceived is

A. Agent program

B. Agent function

C. Percept

D. Percept sequence

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Percept sequence

Percept sequence is explicitly the history of all the agent has perceived, unlike a single current percept. Evidence: PDF 10.

Why each choice is right or wrong

A - Incorrect: Implementation of agent function, not history. See PDF p. 10.

B - Incorrect: Maps history to actions, not history itself. See PDF p. 10.

C - Incorrect: One perceptual instant. See PDF p. 10.

D - Correct: Exact history definition. See PDF p. 10.

Primary note proof

Study-note section AI03

> A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer’s prior instructions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 10 | lines 19-21

> • Percept Sequence − It is the history of all that an agent has perceived till date.
> • Agent Function − It is a map from the precept sequence to an action.
> • An agent is anything that can perceive its environment through sensors and acts upon that

Question sources

Mock exam 1 · Question 2

Question 119 | Q01487

A good cluster is characterized by

A. Equal number of instances

B. Equal distance among cluster members

C. High inter-class similarity

D. High intra-class similarity

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer D: High intra-class similarity

The module groups similar input vectors into clusters. Good within-cluster (intra-class) similarity follows from that grouping principle, with no requirement of equal sizes or equal pairwise distances. Evidence: PDF 75.

Why each choice is right or wrong

A - Incorrect: No equal-size condition given. See PDF p. 75.

B - Incorrect: Similarity does not require identical distances. See PDF p. 75.

C - Incorrect: Between-cluster similarity is not the grouping objective. See PDF p. 75.

D - Correct: Matches similarity inside a group. See PDF p. 75.

Primary note proof

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 75 | lines 19-21

> similar type are combined to form clusters. When a new input pattern is applied, then the neural network gives an output response indicating the class to which the input pattern belongs. There is no feedback from the environment as to what should be the desired output and if it is

Question sources

Mock exam 1 · Question 3

Question 120 | Q01569

A problem in supervised machine learning model due to too much attention to the particular data set it is trained on, causing it to perform poorly on unseen data is ____

A. Under fitting

B. Over fitting

C. Learning curve

D. Variance

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Overfitting.

AI35 directly defines overfitting as fitting training-specific details so performance drops on unseen data. This is the complete failure pattern described in the stem. Underfitting is insufficient fit even on training data; a learning curve is a plot used to inspect performance. Variance may describe sensitivity but does not name the training-versus-generalization failure asked for.

Why each choice is right or wrong

A - Incorrect: Underfitting means the model is too limited or poorly fitted even on the training data.

B - Correct: Overfitting captures training-specific details at the expense of performance on unseen examples.

C - Incorrect: A learning curve plots performance against training size or progress; it is not the failure itself.

D - Incorrect: Variance is related to sensitivity, but overfitting is the precise failure pattern defined by the stem.

Primary note proof

Study-note section AI35

> Overfitting means fitting training-specific details so performance drops on unseen data; underfitting means the model is too limited or poorly fitted even on training data. A learning curve plots performance against training size or training progress. Complexity control, regularization, pruning and more representative data can help.

Study-note section AI-S06

> Support-vector machines include supervised classification and regression methods. They use support vectors in their decision function rather than being defined by a next-state conditional-dependence assumption. In estimation, variance measures how predictions change across different training samples; overfitting instead names poor generalization caused by fitting training-specific details.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 68 | lines 3-5

> tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.
> Thus a learning system is characterized by:

Question sources

Mock exam 3 · Question 1

Question 121 | Q01570

Which general AI search category uses the problem definition and known costs without a problem-specific estimate of remaining cost to a goal?

A. Informed search

B. Uninformed search

C. Greedy best-first search

D. Binary search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Uninformed search

Uninformed search uses the problem definition, including legal actions, goal tests and known action costs, without a domain-specific remaining-cost estimate. Informed methods add such guidance, and greedy best-first uses it directly as h. Binary search is a different ordered-lookup algorithm for sorted sequences. The formerly undefined circular-search distractor is replaced with a meaningful search method.

Why each choice is right or wrong

A - Incorrect: Informed search adds problem-specific guidance beyond the basic problem definition.

B - Correct: Uninformed search uses the problem definition without a heuristic estimate of remaining cost.

C - Incorrect: Greedy best-first explicitly uses h, an estimate of remaining cost.

D - Incorrect: Binary search narrows a sorted sequence by comparisons; it is a specific ordered-lookup algorithm, not the broad AI search category requested.

Primary note proof

Study-note section AI17

A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI-S02

Binary search exploits the order of a sorted sequence: a comparison narrows the remaining candidate interval instead of exploring arbitrary successor states. Array bisection therefore differs from breadth-first or depth-first traversal of a general state-space graph. A storage container such as an array does not by itself specify FIFO or LIFO frontier behavior.

Study-note section AI17

Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 18 | lines 23-25

• The uninformed search does not contain any domain knowledge such as closeness, the location of
the goal. It is also called level order traversal.
• It operates in a brute-force way as it only includes information about how to traverse the tree and

Question sources

Mock exam 3 · Question 2

Question 122 | Q01571

Which named test uses a judge’s inability to distinguish a machine from a human through conversation as its operational criterion?

A. The Chinese room

B. A psychological response-time experiment

C. Turing test

D. Black box testing

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Turing test

The conversational Turing test evaluates whether a judge can reliably distinguish a machine's dialogue from a human's. It supplies an observable behavior criterion rather than proof of understanding. The Chinese-room argument addresses understanding philosophically, response-time experiments test cognitive models empirically, and black-box testing concerns externally visible software functionality. Those are different questions or procedures.

Why each choice is right or wrong

A - Incorrect: The Chinese-room argument concerns whether formal symbol manipulation suffices for understanding; it is not the judge-and-conversation test.

B - Incorrect: A response-time experiment evaluates a cognitive model against human measurements rather than applying conversational indistinguishability.

C - Correct: The conversational Turing test uses this judge-and-hidden-participant criterion.

D - Incorrect: Black-box testing checks external software functionality without inspecting internals; it does not inherently require humanlike conversation.

Primary note proof

Study-note section AI02

> The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Study-note section AI-S07

> Searle's Chinese-room argument challenges whether executing formal symbol-manipulation rules is sufficient for understanding. It is a philosophical argument about understanding, not the conversational judge-and-hidden-participant procedure of the Turing test.

Study-note section AI-S07

> Black-box software testing examines externally visible functionality without inspecting implementation internals. A specification may concern any software behavior; this testing method does not by itself require conversational indistinguishability from a human.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 5 | lines 15-17

> argument Turing put forward the idea of an 'imitation game', in which a human being and a computer would be interrogated under conditions where the interrogator would not know which
> was which, the communication being entirely by textual messages. Turing argued that if the

Question sources

Mock exam 3 · Question 3

Question 123 | Q01612

A learning agent has mainly four conceptual components, which component is responsible for suggesting actions that will lead to new and informative experiences.

A. Problem generator

B. Performance element

C. Learning element

D. Critic

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Problem generator

Problem generator proposes actions producing new informative experience, while the performance element chooses external action, learning element improves it, and critic evaluates it. Evidence: PDF 15.

Why each choice is right or wrong

A - Correct: Matches the explicit exploration role. See PDF p. 15.

B - Incorrect: Selects external action, different component role. See PDF p. 15.

C - Incorrect: Changes performance based on feedback. See PDF p. 15.

D - Incorrect: Evaluates against a performance standard. See PDF p. 15.

Primary note proof

Study-note section AI08

> The performance element selects actions. The learning element changes the agent to improve future performance. The critic evaluates behavior against a performance standard and supplies feedback. The problem generator suggests exploratory actions likely to yield useful experience.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 15 | lines 32-33

> - Problem generator is responsible for suggesting actions that will lead to new and informative experiences.

Question sources

Mock exam 2 · Question 1

Question 124 | Q01613

A learning scheme in which agent learns from a series of rewards & punishments is ____

A. Opinion mining

B. Unsupervised learning

C. Supervised learning

D. Reinforcement learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer D: Reinforcement learning

Rewards and penalties define the module reinforcement scheme. Evidence: PDF 69.

Why each choice is right or wrong

A - Incorrect: Opinion mining is a language application. See PDF p. 69.

B - Incorrect: Input-only pattern discovery lacks reward feedback. See PDF p. 69.

C - Incorrect: Uses paired desired outputs. See PDF p. 69.

D - Correct: Exactly reward/penalty feedback. See PDF p. 69.

Primary note proof

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI37

> Tokenization divides text into processing units; tokens may be words, punctuation or subwords. POS tagging assigns grammatical roles; named-entity recognition identifies people, organizations and locations. Parsing identifies grammatical structure; semantics concerns meaning; discourse links sentences; pragmatics uses context and speaker intent. Sentiment analysis predicts evaluative tone. Ambiguity can occur at every level: “bank” has multiple senses, and “I saw the man with a telescope” has multiple attachments.

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 16-18

reward for each right action and gets a penalty for each wrong action.
The agent learns automatically with these feedbacks and improves its performance. In reinforcement
learning, the agent interacts with the environment and explores it. The goal of an agent is to get the

Question sources

Mock exam 2 · Question 2

Question 125 | Q01614

A learning scheme in which agent learns patterns in the input without any explicit feedback is _____.

A. Supervised learning

B. Unsupervised learning

C. Opinion mining

D. Reinforcement learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Unsupervised learning

The unsupervised network must discover features without feedback about correct outputs. Evidence: PDF 75.

Why each choice is right or wrong

A - Incorrect: Has desired outputs supplied. See PDF p. 75.

B - Correct: Discovers patterns without teacher feedback. See PDF p. 75.

C - Incorrect: Opinion mining is an application, not this learning scheme. See PDF p. 75.

D - Incorrect: Uses evaluative environmental feedback. See PDF p. 75.

Primary note proof

Study-note section AI32

Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI31

Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI37

Tokenization divides text into processing units; tokens may be words, punctuation or subwords. POS tagging assigns grammatical roles; named-entity recognition identifies people, organizations and locations. Parsing identifies grammatical structure; semantics concerns meaning; discourse links sentences; pragmatics uses context and speaker intent. Sentiment analysis predicts evaluative tone. Ambiguity can occur at every level: “bank” has multiple senses, and “I saw the man with a telescope” has multiple attachments.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 75 | lines 21-23

There is no feedback from the environment as to what should be the desired output and if it is correct or incorrect. Hence, in this type of learning, the network itself must discover the patterns and features from the input data, and the relation for the input data over the output.

Question sources

Mock exam 2 · Question 3

Question 126 | SM0010

Suppose you have developed a self-driving car equipped with relevant sensors and cameras. Its sensors are able to detect the environment but not all the aspects. Hence, in which environment is your car
acting?

A. Stochastic

B. Discrete

C. Fully observable

D. Partially observable

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Partially observable

The sensors cannot determine complete state; this is precisely partial observability. It says nothing about transition randomness. Evidence: PDF 13.

Why each choice is right or wrong

A - Incorrect: Stochastic describes transitions, not hidden aspects. See PDF p. 13.

B - Incorrect: Discrete describes state structure, not visibility. See PDF p. 13.

C - Incorrect: Contradicted by incomplete detection. See PDF p. 13.

D - Correct: Matches missing state information. See PDF p. 13.

Primary note proof

Study-note section AI09

> Fully observable means sensors reveal relevant state; partially observable means relevant information is hidden or noisy. Deterministic means state and action determine the next state; stochastic means outcomes involve randomness. Strategic uncertainty about an opponent is distinct from dice-like randomness.

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 13 | lines 14-16

✓ Observable / Partially Observable – If it is possible to determine the complete state of the environment at each time point from the percepts it is observable; otherwise it is only partially observable.

Question sources

MOE Computer Science Exit Exam · Question 3-2

National Computer Science Exit Exam · Question 31

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 193

Question 127 | SM0020

Which quantifier can be used for some portion of the universe?

A. For all (universal)

B. Implies

C. There exists (existential)

D. And

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer C: There exists (existential)

Some or at least one is existential quantification; universal applies to all. Evidence: PDF 49.

Why each choice is right or wrong

A - Incorrect: All instances rather than some. See PDF p. 49.

B - Incorrect: A connective, not a quantifier. See PDF p. 49.

C - Correct: Matches some/at least one. See PDF p. 49.

D - Incorrect: A connective, not a quantifier. See PDF p. 49.

Primary note proof

Study-note section AI26

First-order logic adds objects, predicates, functions and quantified object variables. Human(a) is a property; Likes(a,b) a relation; father(a) a function term. Propositional logic has no object quantifiers, but its truth still depends on the assigned atom values.

Study-note section AI25

Propositional atoms represent whole true-or-false claims. ¬P negates; P∧Q requires both; P∨Q requires at least one; P→Q is false only when P is true and Q false; P↔Q requires matching truth values. Thus P→Q ≡ ¬P∨Q. De Morgan gives ¬(P∧Q) ≡ ¬P∨¬Q.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 49 | lines 11-13

• Existential quantifier, (for some, at least one).
1. Universal Quantifier: - is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing.

Question sources

MOE Computer Science Exit Exam · Question 5-4

National Computer Science Exit Exam · Question 21

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 194

Question 128 | SM0037

Which one of the following is a step in problem solving that lists the actions and states that the agent has to consider provided a goal?

A. Problem formulation

B. Execute

C. Search

D. Goal formulation

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer A: Problem formulation.

AI11 says the goal is formulated first, then the problem specifies initial state, legal actions, transitions, goal test and action costs. Given a goal, listing the relevant actions and states belongs to that second step. Search chooses a sequence within the formulated problem, and execution applies it; neither defines the alternatives initially.

Why each choice is right or wrong

A - Correct: Problem formulation specifies the states, available actions and model the agent must consider for its goal.

B - Incorrect: Execution applies a selected action sequence after formulation and search.

C - Incorrect: Search selects a solution using the already specified states, actions, model and goal test.

D - Incorrect: Goal formulation chooses the objective; the stem already provides a goal before listing actions and states.

Primary note proof

Study-note section AI11

> First formulate the goal; then formulate the problem by specifying initial state, legal actions, transition model, goal test, and action costs. Search selects a sequence; execution applies it. A state abstracts away irrelevant world details.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 18 | lines 16-18

> Actions: It gives the description of all the available actions to the agent.
> Transition model: A description of what each action do, can be represented as a transition model.
> Path Cost: It is a function which assigns a numeric cost to each path.

Question sources

MOE Computer Science Exit Exam · Question 11-3

National Computer Science Exit Exam · Question 98

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 195

Question 129 | SM0061

Which term names the entire ordered history of observations an agent has received so far?

A. Performance measure

B. Percept sequence

C. Knowledge

D. Action

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Percept sequence

A percept is one observation, while a percept sequence is the entire observation history. A performance measure defines success instead of storing observations. Knowledge may include prior facts and inferences beyond the observation history, and an action is selected behavior rather than observed input. Replacing the overly broad perception option supplies the precise established term without changing the agent-history concept.

Why each choice is right or wrong

A - Incorrect: A performance measure defines success, not what the agent has observed.

B - Correct: A percept sequence is the entire observation history.

C - Incorrect: Knowledge includes prior and inferred information, not precisely the observation history.

D - Incorrect: An action is selected behavior rather than the agent's accumulated observations.

Primary note proof

Study-note section AI03

> A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer's prior instructions.

Study-note section AI04

> A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 10 | lines 19-21

> • Percept Sequence − It is the history of all that an agent has perceived till date.
> • Agent Function − It is a map from the precept sequence to an action.
> • An agent is anything that can perceive its environment through sensors and acts upon that

Question sources

MOE Computer Science Exit Exam · Question 17-4

National Computer Science Exit Exam · Question 51

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 198

Question 130 | SM0211

Every training example has an input and its desired target output; there is no extra unlabeled training pool or interaction reward. Which learning regime fits this setup?

A. Semi-supervised learning

B. Unsupervised learning

C. Supervised learning

D. Reinforcement learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: Supervised learning

The training data consist entirely of input-target pairs, so the task is supervised learning. The desired target provides the learning signal. Semi-supervised learning would add an unlabeled pool, unsupervised learning would seek structure without targets, and reinforcement learning would use rewards from interaction. Stating the complete data setup distinguishes the regimes without implying that any use of labels excludes semi-supervised methods.

Why each choice is right or wrong

A - Incorrect: Semi-supervised learning requires additional unlabeled training data alongside labeled examples, absent here.

B - Incorrect: Unsupervised learning discovers structure without the supplied target labels.

C - Correct: Supervised learning fits the mapping from the labeled input-target pairs described.

D - Incorrect: Reinforcement learning uses evaluative interaction rewards, not the described desired-output pairs.

Primary note proof

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

> Semi-supervised learning combines some labels with additional unlabeled data. Active learning chooses informative cases for an oracle to label; it still uses labels. “Exploratory learning” is not an exclusive standard synonym: it can mean unsupervised exploration or active data acquisition, so define the feedback. A task named summarization or time-series analysis is not automatically supervised; its training targets determine that.

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 68 | lines 25-25

⇒ Supervised learning implies we are given a set of (x, y) pairs by a "teacher." The system is supplied

Introduction to Artificial Intelligence module | PDF page 69 | lines 2-4

with a set of training examples consisting of inputs and corresponding outputs, and is required to
discover the relation or mapping between then, e.g. as a series of rules, or a neural network. Agent
learns a function from observing example input-output pairs

Question sources

Computer Science Exit Exam · Sene 2016 · Question 4-1

Computer Science Exit Exam practice collection · Question 207

Question 131 | SM0222

The simple strategy in which the root node is expanded first, then all successors are expanded next, layer by layer, refers to:

A. Greedy best-first search

B. Depth-first search (DFS)

C. Depth limited search (DLS)

D. Breadth-first search (BFS)

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer D: Breadth-first search (BFS)

Expanding every node at one level before the next is the stated BFS order. Evidence: PDF 19.

Why each choice is right or wrong

A - Incorrect: Ranks by estimated goal distance. See PDF p. 19.

B - Incorrect: Follows a path deeply before siblings. See PDF p. 19.

C - Incorrect: Restricts DFS to a bound. See PDF p. 19.

D - Correct: Exact layer-order expansion. See PDF p. 19.

Primary note proof

Study-note section AI13

BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Study-note section AI16

DLS runs DFS to depth l and reports cutoff separately from failure. It misses goals deeper than l. Iterative deepening runs DLS with limits 0,1,2,...; for finite b it finds the shallowest solution with O(b^d) time and O(bd) frontier space, optimal for equal step costs. Repeated shallow work is small compared with the deepest level.

Study-note section AI17

Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 19 | lines 25-26

• BFS algorithm starts searching from the root node of the tree and expands all successor node at the
current level before moving to nodes of next level.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 7-3

Computer Science Exit Exam practice collection · Question 206

Question 132 | SM0253

Which statement is not a correct characterization of reasoning?

A. Using knowledge to derive conclusions that can guide what a system does.

B. A process of thinking, logically arguing, and drawing inference

C. Theory of blind searching

D. The act of deriving a conclusion from certain premises using a given methodology

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Theory of blind searching

Reasoning uses knowledge and a specified methodology to derive conclusions from premises. Logical argument and inference are therefore appropriate descriptions. Blind search is instead a search strategy that explores alternatives without heuristic guidance; it does not define reasoning as a whole. Clarifying the incomplete knowledge-system option into a claim about deriving conclusions leaves the intended search-versus-inference distinction as the unique exception.

Why each choice is right or wrong

A - Incorrect: Using knowledge to derive conclusions is a valid characterization, so it is not the exception.

B - Incorrect: Logical argument and drawing inferences describe reasoning, so this is not a false characterization for the negative question.

C - Correct: Blind search is one search strategy rather than a definition of reasoning.

D - Incorrect: Deriving a conclusion from premises using a methodology describes reasoning and does not satisfy the negative question.

Primary note proof

Study-note section AI24

> Knowledge representation encodes facts, objects, relations, events and rules so a machine can use them. Syntax defines legal expressions; semantics defines their meaning. A knowledge base stores assertions; an inference system derives consequences. Declarative knowledge says what is true; procedural knowledge specifies how to act.

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 53 | lines 18-20

> - Reasoning - is the mental process of deriving logical conclusion and making predictions from available knowledge, facts, and beliefs. "Reasoning is a way to infer facts from existing data." It is a general process of thinking rationally, to find valid conclusions. In artificial intelligence, the

Question sources

Computer Science Exit Exam · Sene 2016 · Question 17-2

Computer Science Exit Exam practice collection · Question 208

Question 133 | SM0261

Which reasoning technique applies valid inference rules so that true premises guarantee a true conclusion?

A. Abductive reasoning

B. Analogical reasoning

C. Deductive reasoning

D. Inductive reasoning

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer C: Deductive reasoning

Deductive reasoning uses truth-preserving inference: when the premises are true and the rule is valid, the conclusion must be true. Rules in a broad sense are not sufficient to identify deduction, since many reasoning methods use procedures. Induction generalizes from cases, abduction proposes explanations, and analogy transfers conclusions across similarities; these do not supply the stated logical guarantee.

Why each choice is right or wrong

A - Incorrect: Abduction proposes plausible explanatory hypotheses that can still be false.

B - Incorrect: Analogy transfers a conclusion from a similar case without a truth-preservation guarantee.

C - Correct: Valid deductive inference preserves truth from premises to conclusion.

D - Incorrect: Induction generalizes beyond the observed cases and remains revisable.

Primary note proof

Study-note section AI28

> Deduction applies truth-preserving rules: all people are mortal; Ada is a person; therefore Ada is mortal. Induction generalizes observations to a broader rule and can be overturned by new

examples. Abduction proposes a plausible explanation: wet ground suggests rain, although sprinklers could explain it. Analogy transfers a pattern between similar situations.

Study-note section AI28

Classical logical entailment is monotonic: adding premises preserves previous consequences. Default reasoning can be nonmonotonic: infer a bird usually flies, then withdraw this when told it is a penguin. Common-sense defaults are useful with incomplete knowledge, but are not guaranteed truths. Blind search is a search strategy, not a definition of reasoning.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 53 | lines 23-25

A. Deductive reasoning - it deducing new information from logically related known information.
o It is the form of valid reasoning, which means the argument's conclusion must be true when the
premises are true.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 20-1

Computer Science Exit Exam practice collection · Question 204

Question 134 | SM0269

For A* tree search with an admissible heuristic and positive step costs, what is the intended benefit of using the heuristic alongside accumulated edge costs?

A. To seek a least-cost path while potentially reducing expansions through heuristic guidance.

B. To find the path with the maximum distance from the start to the goal node

C. To find paths with non-optimal searches from node to the goal node

D. To find the path with the maximum cost from the start to the goal node

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: To seek a least-cost path while potentially reducing expansions through heuristic guidance.

A* adds accumulated cost g to an admissible remaining-cost estimate h. Under the stated tree-search conditions, this supports least-cost solutions, while useful heuristic guidance can reduce expansions. It does not guarantee a speed improvement for every problem or heuristic. Maximizing distance or cost is a different objective, and deliberately seeking nonoptimal paths is not the benefit described.

Why each choice is right or wrong

A - Correct: An admissible remaining-cost estimate supports optimal tree search and can guide exploration toward useful candidates.

B - Incorrect: A* minimizes total path cost; it does not seek maximum path length.

C - Incorrect: Nonoptimality is not the purpose of admissible A* tree search.

D - Incorrect: A* seeks minimum, not maximum, total cost.

Primary note proof

Study-note section AI18

A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI18

Admissibility suffices for optimal A* tree search; graph search that never reopens CLOSED states normally requires consistency. With an admissible inconsistent heuristic, reopen states when g improves. Test a goal when removed at minimum f. Completeness also needs finite branching and positive lower-bounded costs in infinite spaces. Worst-case time and memory can be exponential; no heuristic universally guarantees speed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 27 | lines 2-4

• A* search algorithm finds the shortest path through the search space using the heuristic function.
This search algorithm expands less search tree and provides optimal result faster. A* algorithm is
similar to UCS except that it uses g(n)+h(n) instead of g(n).

Introduction to Artificial Intelligence module | PDF page 28 | lines 14-19

• Optimal: A* search algorithm is optimal if it follows below two conditions:
o Admissible: the first condition requires for optimality is that h(n) should be an admissible heuristic for A* tree search. An admissible heuristic is optimistic in nature.
o Consistency: Second required condition is consistency for only A* graph-search.
• If the heuristic function is admissible, then A* tree search will always find the least cost path.
• Time Complexity: The time complexity of A* search algorithm depends on heuristic function, and

Question sources

Computer Science Exit Exam · Sene 2016 · Question 22-3

Computer Science Exit Exam practice collection · Question 209

Question 135 | SM0299

What role does machine learning play in understanding human perception?

A. Machine learning helps simulate human perception through neural networks

B. Machine learning has no relevance to understanding human perception

C. Machine learning is not capable of analyzing perceptual data

D. Machine learning only focuses on logical reasoning and decision-making

SYSTEMS

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer A: Machine learning helps simulate human perception through neural networks

ANNs model brain-inspired processing and learn pattern recognition; combined with p67 recognition applications this supports the perception-modeling role. Evidence: PDF 71, 71, 67.

Why each choice is right or wrong

A - Correct: Matches brain-inspired models applied to recognition. See PDF p. 71, 71, 67.

B - Incorrect: Contradicts documented pattern-recognition applications. See PDF p. 71, 71, 67.

C - Incorrect: Contradicts recognition/classification capabilities. See PDF p. 71, 71, 67.

D - Incorrect: Machine learning also handles sensory patterns, not only logic. See PDF p. 71, 71, 67.

Primary note proof

Study-note section AI34

> Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Study-note section AI39

> Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 71 | lines 6-8

> An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way
> biological nervous systems, such as the brain, process information. The key element of this paradigm
> is the novel structure of the information processing system.

Introduction to Artificial Intelligence module | PDF page 71 | lines 11-13

> configured for a specific application, such as pattern recognition or data classification, through a learning
> process. Learning in biological systems involves adjustments to the synaptic connections that exist
> between the neurons. This is true of ANNs as well. ANNs can process information at a great speed

Introduction to Artificial Intelligence module | PDF page 67 | lines 30-31

> • Speech/image/video (pattern) recognition
> • Adaptive control

Question sources

Computer Science Exit Exam · Sene 2016 · Question 32-2

Computer Science Exit Exam practice collection · Question 210

Question 136 | SS100

An informed search keeps one priority frontier ordered by g(n)+h(n), permanently closes states, and tests goals on removal. With a consistent heuristic, finite branching and positive lower-bounded step costs, which algorithm is this?

A. Iterative Deepening A* Search

B. Greedy Best-First Search

C. Uniform Cost Search

D. A* Search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer D: A* Search

A* uses a best-first priority frontier ordered by g+h. Consistency provides the standard guarantee for graph search that permanently closes states, with the stated cost and goal-test conditions. IDA* also uses f values but organizes repeated depth-first threshold searches instead. Greedy search omits g, and UCS omits h. The frontier organization distinguishes the algorithms rather than a worst-case complexity shared by several methods.

Why each choice is right or wrong

A - Incorrect: IDA* repeats depth-first searches under increasing f thresholds rather than retaining this single priority frontier.

B - Incorrect: Greedy best-first uses h alone and does not have the stated cost-aware priority.

C - Incorrect: UCS uses g alone, without the heuristic term.

D - Correct: A* uses g+h; consistency supports optimality when graph states are closed without reopening.

Primary note proof

Study-note section AI18

A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI18

Admissibility suffices for optimal A* tree search; graph search that never reopens CLOSED states normally requires consistency. With an admissible inconsistent heuristic, reopen states when g improves. Test a goal when removed at minimum f. Completeness also needs finite branching and positive lower-bounded costs in infinite spaces. Worst-case time and memory can be exponential; no heuristic universally guarantees speed.

Study-note section AI16

Bidirectional search grows from start and goal until frontiers connect. With suitable reverse actions, meeting tests and equal-cost BFS, time and space may fall to O(b^(d/2)). Weighted variants require a sound stopping bound; the first intersection need not be cheapest. IDA* deepens an f=g+h threshold rather than only a depth limit.

Study-note section AI17

Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI15

UCS uses a priority queue ordered by actual accumulated path cost g(n).

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 28 | lines 19-21

• Time Complexity: The time complexity of A* search algorithm depends on heuristic function, and
the number of nodes expanded is exponential to the depth of solution d. So the time complexity

is
O(bd), where b is the branching factor.

Introduction to Artificial Intelligence module | PDF page 28 | lines 14-19

• Optimal: A* search algorithm is optimal if it follows below two conditions:
o Admissible: the first condition requires for optimality is that h(n) should be an admissible heuristic for A* tree search. An admissible heuristic is optimistic in nature.
o Consistency: Second required condition is consistency for only A* graph-search.
• If the heuristic function is admissible, then A* tree search will always find the least cost path.
• Time Complexity: The time complexity of A* search algorithm depends on heuristic function, and

Introduction to Artificial Intelligence module | PDF page 26 | lines 17-18

o This algorithm is not optimal.
• Time Complexity: The worst case time complexity of Greedy best first search is O(bm).

Question sources

Computer Science model exam collection · Question 7

Question 137 | SS143

Which general term denotes an entity that perceives its environment through sensors and acts on it through effectors?

A. API

B. Expert System

C. Intelligence

D. Agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Agent

An agent is defined by perceiving an environment and acting on it through sensors and effectors. The acting mechanism may be physical or software-based. An API can expose operations used by an agent, and an expert system can contribute domain reasoning, but neither names the general entity defined here. Intelligence is a capability rather than the entity itself.

Why each choice is right or wrong

A - Incorrect: An API exposes software operations; it may support an agent's actions but is not the general perceiving-and-acting entity.

B - Incorrect: An expert system is a specialized knowledge-based application, not the general term for every sensing-and-acting entity.

C - Incorrect: Intelligence is a capability, not the name of the defined entity.

D - Correct: Agent is the general term for an entity with this perception-and-action interaction.

Primary note proof

Study-note section AI03

> An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Study-note section AI-S06

> An application programming interface (API) exposes defined operations through which software requests functionality. It can be an interface used by an agent, but the interface alone is not the general entity that perceives an environment and selects actions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 10 | lines 21-23

> • An agent is anything that can perceive its environment through sensors and acts upon that environment through effectors. An agent perceives its environment through sensors.
> • The complete set of inputs at a given time is called a percept.

Question sources

Computer Science model exam collection · Question 50

Question 138 | SS171

A new dataset has no target labels. The objective is to discover its groups or structure, with neither interaction rewards nor reuse of a pretrained model. Which learning regime applies?

A. Reinforcement learning

B. Unsupervised learning

C. Transfer learning

D. Supervised learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Unsupervised learning

The objective is structure discovery without target labels, which defines unsupervised learning. The stem separately excludes action-reward interaction and reuse of a pretrained model. Those exclusions distinguish reinforcement feedback and transfer learning from this task. Transfer learning describes a reuse strategy rather than a label-availability regime and can be combined with different learning objectives.

Why each choice is right or wrong

A - Incorrect: Reinforcement learning requires a reward-based interaction objective, which is excluded.

B - Correct: Unsupervised learning discovers structure in data without supplied target labels.

C - Incorrect: Transfer learning concerns reuse of prior learned representations or models; no such reuse is present.

D - Incorrect: Supervised learning requires target outputs paired with the inputs.

Primary note proof

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI-S06

> Transfer learning reuses representations or a model learned on an earlier task for a new task, for example adapting pretrained image features to new image classes. This is a reuse strategy, not a specification that the new task has no labels or rewards.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 7-9

⇒ Unsupervised learning means we are only given the x‘s. In either case, the goal is to estimate f. The system is supplied with a set of training examples consisting only of inputs and is required to discover for itself what appropriate outputs should be,

Question sources

Computer Science model exam collection · Question 78

Question 139 | SS175

One of the following statement is correct about Artificial Intelligence (Al)?

A. It refers to the development of machines that can perform tasks only in a controlled laboratory setting.

B. It refers to the development of machines that can only perform simple, repetitive tasks.

C. Itrefers to the development of machines that can only perform physical tasks.

D. It refers to the development of machines that can perform tasks that typically require human intelligence.

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer D: It refers to the development of machines that can perform tasks that typically require human intelligence.

AI designs intelligence in artificial devices and includes language, learning, reasoning and real-world action, not the restrictive only-claims. Evidence: PDF 4, 6.

Why each choice is right or wrong

A - Incorrect: Real-world robots contradict laboratory-only. See PDF p. 4, 6.

B - Incorrect: Reasoning and complex problem solving exceed repetition. See PDF p. 4, 6.

C - Incorrect: Language/reasoning are not merely physical. See PDF p. 4, 6.

D - Correct: Matches design of artificial intelligence. See PDF p. 4, 6.

Primary note proof

Study-note section AI01

Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI39

Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector.

Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 4 | lines 26-28

is concerned with the design of intelligence in an artificial device. The term was coined by McCarthy
in 1956. There are two ideas in the definition. i.e Intelligence, and Artificial device.
What is intelligence?

Introduction to Artificial Intelligence module | PDF page 6 | lines 12-21

1.3Intelligent behaviour
This discussion brings us back to the question of what constitutes intelligent behavior. Some ofthese
tasks and applications are:
• Perception involving image recognition and computer vision
• Reasoning
• Learning
• Understanding language involving natural language processing, speech processing
• Solving problems
• Robotics
• Thinking

Question sources

Computer Science model exam collection · Question 82

Question 140 | SS187

A techniques for generating plans with conditionals and loops are almost identical to those for generating programs from logical specifications is
called__

A. Automatic learning

B. Automatic goal testing

C. Automatic monitoring

D. Automatic programming

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: Automatic programming

Automatic programming, also called program synthesis, constructs a program that satisfies a specification. Generating conditional plans or loops is analogous because the output must implement the required behavior. Learning changes performance through experience, goal testing checks whether a candidate succeeds, and monitoring checks execution. None of those by itself names the specification-to-program construction task.

Why each choice is right or wrong

A - Incorrect: Learning improves performance through experience rather than specifically synthesizing a program from logical requirements.

B - Incorrect: A goal test checks whether a candidate meets the goal; it does not generate a program from its specification.

C - Incorrect: Monitoring checks an executing system rather than generating a program from a specification.

D - Correct: Automatic programming or program synthesis constructs a program meeting a specification, including conditional or looping behavior.

Primary note proof

Study-note section AI11

> In route finding, states are cities, actions are roads, successors are neighboring cities, and path cost is accumulated distance or time. A solution is a legal path ending at a goal; an optimal solution has minimum cost. Automatic programming or program synthesis searches for a program satisfying a specification; conditional plans and loops require richer representations than a fixed action list.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Study-note section AI11

> First formulate the goal; then formulate the problem by specifying initial state, legal actions, transition model, goal test, and action costs. Search selects a sequence; execution applies it. A state abstracts away irrelevant world details.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 64 | lines 6-8

> • Backward Chaining
> o With this strategy, an expert system finds out the answer to the question, —Why this happened?‖

o On the basis of what has already happened, the Inference Engine tries to find out which conditions

Question sources

Computer Science model exam collection · Question 94

Question 141 | AM0624

Which choices describe broad intelligence-related goals of AI rather than implementation means or hardware capacity? Select all that apply.

A. Emulating human intelligence

B. Creating machines that can think and learn

C. Developing advanced algorithms

D. Enhancing computer processing power

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer A, B: Emulating human intelligence; Creating machines that can think and learn

AI can aim to emulate aspects of human intelligence or to build systems that reason and learn. Those are compatible broad goals, although different AI approaches evaluate humanlike behavior and rational performance differently. Advanced algorithms are means that may serve many fields, while additional processing power is hardware capacity rather than intelligence itself. The distinction therefore permits both intelligence-related objectives.

Why each choice is right or wrong

A - Correct: Humanlike thought and behavior are established AI approaches and fit this broad goal.

B - Correct: Building systems with intelligent reasoning and learning is another supported broad AI objective.

C - Incorrect: Advanced algorithms are implementation means, not the defining intelligence goal.

D - Incorrect: More processing power alone does not establish intelligent behavior or learning.

Primary note proof

Study-note section AI01

Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI43

> Face recognition is both an applied task and a narrow capability. Broad machine-intelligence goals can include emulating human intelligence and building systems that learn. Humanlike problem-solving alone does not uniquely select expert systems. Python is widely used, but C++ and Java also implement AI; “commonly used” does not isolate one language. McCarthy is associated with naming AI, not its sole invention.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 8 | lines 9-11

> General AI goal
> • Replicate human intelligent
> • Solve knowledge intensive task

Introduction to Artificial Intelligence module | PDF page 8 | lines 14-16

> Engineering based AI goal
> • Develop concepts, theory and practice of building intelligent machines
> • Emphasis is on system building

Question sources

Computer Science MCQ practice collection · Question 624

Question 142 | AM0625

What is knowledge representation in artificial intelligence?

A. The process of storing information in a computer

B. The ability of machines to reason and make decisions

C. The study of how humans acquire knowledge

D. The process of encoding knowledge for machine use

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer D: The process of encoding knowledge for machine use

Knowledge representation organizes and formalizes knowledge in a machine-usable knowledge base. Storage alone omits the encoding/formalization purpose. Evidence: PDF 62.

Why each choice is right or wrong

A - Incorrect: Too broad: mere storage need not represent usable knowledge. See PDF p. 62.

B - Incorrect: Reasoning uses representations but is not representation itself. See PDF p. 62.

C - Incorrect: Human acquisition is a different subject. See PDF p. 62.

D - Correct: Best match to formal machine-usable encoding. See PDF p. 62.

Primary note proof

Study-note section AI24

> Knowledge representation encodes facts, objects, relations, events and rules so a machine can use them. Syntax defines legal expressions; semantics defines their meaning. A knowledge base stores assertions; an inference system derives consequences. Declarative knowledge says what is true; procedural knowledge specifies how to act.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 62 | lines 27-27

> o It is the method used to organize and formalize the knowledge in the knowledge base.

Question sources

Computer Science MCQ practice collection · Question 625

Question 143 | AM0626

Which representation system matches IF-THEN production rules against working memory and applies a selected rule to derive a conclusion or action?

A. Semantic networks

B. Neural networks

C. Production systems

D. Bayesian networks

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: Production systems

A production system uses IF-THEN rules whose conditions are matched against working memory; an applicable rule is selected and applied. Semantic networks encode concept relationships, neural networks learn weighted representations, and Bayesian networks encode probabilistic dependencies. All can support reasoning in broader systems, so the distinctive feature here is the explicit production-rule cycle rather than inference in general.

Why each choice is right or wrong

A - Incorrect: Semantic networks represent concepts and labeled relationships, not this working-memory production-rule cycle.

B - Incorrect: Neural networks compute through weighted units and learned representations rather than the specified rule-matching architecture.

C - Correct: Production systems match condition-action or condition-conclusion rules against working memory.

D - Incorrect: Bayesian networks represent conditional probabilistic dependencies rather than the stated production-rule mechanism.

Primary note proof

Study-note section AI24

A semantic network connects concept nodes by labeled relations such as is-a. Frames describe stereotyped entities through slots, values and defaults. Production systems use IF conditions THEN conclusions or actions, matching rules against working memory and selecting among applicable rules. A graph suits relationships such as frequently bought together; frames suit attributes. No representation is always best for recommendation without task requirements.

Study-note section AI29

A Bayesian network is a directed acyclic graph with conditional probability tables; its joint distribution factors as the product of each node's probability given its parents. A Markov model assumes the next state depends on the current state rather than the entire past. These represent uncertainty; fuzzy membership represents a different issue.

Study-note section AI34

Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 63 | lines 20-22

o Applies rules repeatedly to the facts, which are obtained from earlier rule application.
o Adds new knowledge into the knowledge base if required.
o Resolves rules conflict when multiple rules are applicable to a particular case

Introduction to Artificial Intelligence module | PDF page 64 | lines 5-6

and production rule systems.
• Backward Chaining

Question sources

Computer Science MCQ practice collection · Question 626

Question 144 | AM0627

Which reasoning technique involves drawing conclusions from specific observations or examples?

A. Deductive reasoning

B. Inductive reasoning

C. Abductive reasoning

D. Probabilistic reasoning

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer B: Inductive reasoning

Specific facts generalized into a conclusion define induction. The conclusion is supported rather than logically guaranteed by the finite examples. Evidence: PDF 53.

Why each choice is right or wrong

A - Incorrect: Typically general premises to specific conclusion. See PDF p. 53.

B - Correct: Exact specific-to-general pattern. See PDF p. 53.

C - Incorrect: Seeks most likely explanation for observations. See PDF p. 53.

D - Incorrect: Uses probability to express uncertainty, not this defining direction. See PDF p. 53.

Primary note proof

Study-note section AI28

> Deduction applies truth-preserving rules: all people are mortal; Ada is a person; therefore Ada is mortal. Induction generalizes observations to a broader rule and can be overturned by new examples. Abduction proposes a plausible explanation: wet ground suggests rain, although sprinklers could explain it. Analogy transfers a pattern between similar situations.

Study-note section AI29

> Probability describes uncertainty about events. $P(\neg A)=1-P(A)$; $P(A|B)=P(A\cap B)/P(B)$, for $P(B)>0$. Bayes gives $P(H|E)=P(E|H)P(H)/P(E)$: posterior equals likelihood times prior divided by evidence probability. For mutually exclusive exhaustive hypotheses, $P(E)=\Sigma_i P(E|Hi)P(Hi)$.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 53 | lines 32-33

o It starts with the series of specific facts or data and reaches to a general statement or conclusion.

Question sources

Computer Science MCQ practice collection · Question 627

Question 145 | AM0628

A classifier is trained on labeled examples and predicts by following attribute tests from a root to a class-labeled leaf. Which learning technique matches this representation?

A. Reinforcement learning

B. Unsupervised learning

C. Deep learning

D. Decision tree learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer D: Decision tree learning

A decision tree predicts by testing an attribute at each internal node, following a branch and returning the leaf prediction. Labeled training makes this a supervised classification task. Deep learning can also be supervised, so supervision alone would not distinguish it. The representation in the stem provides the distinction; reinforcement and unsupervised learning describe different feedback regimes.

Why each choice is right or wrong

A - Incorrect: Reinforcement learning specifies environmental reward feedback rather than this attribute-test representation.

B - Incorrect: Unsupervised learning seeks structure without supplied target labels.

C - Incorrect: Deep learning uses multiple neural representation layers; it can be supervised but is not the attribute-test-and-leaf model described.

D - Correct: Decision tree learning produces attribute-test branches with predictions at leaves.

Primary note proof

Study-note section AI31

Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI31

A decision tree tests an attribute at each internal node, follows the corresponding branch and predicts at a leaf. ID3 chooses high information gain: $H(S)=-\Sigma p_i \log_2 p_i$; gain is parent entropy minus weighted child entropy. Gini impurity is $1-\Sigma p_i^2$, zero for a pure class. Pruning limits overfitting. Neural networks, support-vector machines and trees can all be supervised.

Study-note section AI43

ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Study-note section AI34

Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 21-23

A. Decision tree algorithm
⇒ It is a classification and prediction tool having a tree like structure, where each internal node denotes
a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal

Introduction to Artificial Intelligence module | PDF page 69 | lines 5-7

o Classification
o Regression
⇒ Unsupervised learning means we are only given the x‘s. In either case, the goal is to estimate f.

Question sources

Computer Science MCQ practice collection · Question 628

Question 146 | AM0629

What is the primary objective of natural language processing (NLP) in artificial intelligence?

A. Understanding and generating human language by machines

B. Analyzing and processing large datasets

C. Enhancing computer hardware capabilities

D. Simulating human-like decision-making

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer A: Understanding and generating human language by machines

Understanding/interpretation is defined in NLP, and NLG turns data into language, supporting both understanding and generation. Evidence: PDF 79, 79.

Why each choice is right or wrong

A - Correct: Matches understanding plus NLG. See PDF p. 79, 79.

B - Incorrect: Large-data processing is broader than language. See PDF p. 79, 79.

C - Incorrect: Not a hardware discipline. See PDF p. 79, 79.

D - Incorrect: Too broad; omits the defining language objective. See PDF p. 79, 79.

Primary note proof

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 79 | lines 4-6

> and Artificial Intelligence. It is the technology that is used by machines to understand, analyze, manipulate, and interpret human's languages. It helps developers to organize knowledge for performing tasks such as translation, automatic summarization, Named Entity Recognition (NER),

Introduction to Artificial Intelligence module | PDF page 79 | lines 31-33

> - Natural Language Generation (NLG) acts as a translator that converts the computerized data into natural language representation. It mainly involves Text planning, Sentence planning, and Text Realization.

Question sources

Computer Science MCQ practice collection · Question 629

Question 147 | AM0630

Which model family assumes that the distribution of the next state depends on the current state rather than the entire earlier history?

A. Decision trees

B. Markov models

C. Support vector machines

D. Reinforcement learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Markov models

The Markov assumption makes the current state sufficient for the next-state distribution, rather than requiring the full preceding history. That identifies the model family. Decision trees and support-vector machines are prediction methods with other defining structures. Reinforcement learning is a reward-based learning framework that can use a Markov model, but it is not itself the particular conditional-dependence assumption.

Why each choice is right or wrong

A - Incorrect: Decision trees are defined by attribute tests and leaf predictions rather than this conditional transition assumption.

B - Correct: A Markov model conditions the next state on the current state instead of the full previous history.

C - Incorrect: Support-vector machines use support vectors for prediction; a next-state conditional-dependence assumption does not define the family.

D - Incorrect: Reinforcement learning specifies improvement from rewards and may use different model families.

Primary note proof

Study-note section AI29

> A Bayesian network is a directed acyclic graph with conditional probability tables; its joint distribution factors as the product of each node's probability given its parents. A Markov model assumes the next state depends on the current state rather than the entire past. These represent uncertainty; fuzzy membership represents a different issue.

Study-note section AI31

> A decision tree tests an attribute at each internal node, follows the corresponding branch and predicts at a leaf. ID3 chooses high information gain: $H(S)=-\Sigma p_i \log_2 p_i$; gain is parent entropy minus weighted child entropy. Gini impurity is $1-\Sigma p_i^2$, zero for a pure class. Pruning limits overfitting. Neural networks, support-vector machines and trees can all be supervised.

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI-S06

Support-vector machines include supervised classification and regression methods. They use support vectors in their decision function rather than being defined by a next-state conditional-dependence assumption. In estimation, variance measures how predictions change across different training samples; overfitting instead names poor generalization caused by fitting training-specific details.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 55 | lines 27-29

• Probabilistic reasoning is a way of knowledge representation where we apply the concept of probability to indicate the uncertainty in knowledge.
• In probabilistic reasoning, we combine probability theory with logic to handle the uncertainty.

Question sources

Computer Science MCQ practice collection · Question 630

Question 148 | AM0631

Which learning regime is defined by an agent trying actions, receiving environmental rewards and improving a policy to maximize expected cumulative return?

A. Reinforcement learning

B. Unsupervised learning

C. Deep learning

D. Genetic algorithms

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: Reinforcement learning

Reinforcement learning improves an action policy through rewards received during interaction, with expected cumulative return as the objective. Generic trial and error is too broad because evolutionary search also evaluates and improves candidates. Genetic algorithms are identified by population selection and variation, while deep learning identifies architecture. Unsupervised learning concerns unlabeled structure rather than this sequential reward objective.

Why each choice is right or wrong

A - Correct: Reinforcement learning is defined by this action-reward policy-improvement objective.

B - Incorrect: Unsupervised learning discovers unlabeled structure rather than defining cumulative-return optimization.

C - Incorrect: Deep learning specifies multilayer neural representations and can be used within several learning regimes.

D - Incorrect: Genetic algorithms improve candidate populations using selection and genetic variation; trial and error alone does not make them the stated action-reward learning regime.

Primary note proof

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI43

> ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Study-note section AI20

> A genetic algorithm maintains a population, evaluates fitness, selects parents, applies crossover and mutation, and repeats replacement. It searches without guaranteeing global optimality. Means-ends analysis selects a difference between current and goal states, chooses an operator reducing it, and recursively solves its preconditions. Neither method is simply A* running backward.

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 69 | lines 18-20

> learning, the agent interacts with the environment and explores it. The goal of an agent is to get the
> most reward points, and hence, it improves its performance. The robotic dog, which automatically
> learns the movement of his arms, is an example of Reinforcement learning.

Question sources

Computer Science MCQ practice collection · Question 631

Question 149 | AM0632

Which type of learning algorithm aims to mimic the structure and function of the human brain?

A. Supervised learning

B. Unsupervised learning

C. Deep learning

D. Reinforcement learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: Deep learning, as a multilayer brain-inspired neural architecture.

AI34 calls ANNs brain-inspired abstractions and describes deep learning as using multiple representation layers. AI43 stresses that deep models can use supervised, unsupervised or reinforcement objectives. Thus deep learning is the offered architecture-based answer, while the other choices classify feedback regimes rather than requiring brain-inspired structure. This is inspiration, not a claim of literal brain simulation.

Why each choice is right or wrong

A - Incorrect: Supervised learning specifies labeled targets and can use non-neural models such as trees or SVMs.

B - Incorrect: Unsupervised learning specifies unlabeled structure discovery and does not require a neural architecture.

C - Correct: Deep learning uses multilayer neural representations within the brain-inspired ANN family.

D - Incorrect: Reinforcement learning specifies reward-based interaction and need not use brain-inspired neural structure.

Primary note proof

Study-note section AI34

> Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Study-note section AI43

ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 71 | lines 7-9

biological nervous systems, such as the brain, process information. The key element of this paradigm
is the novel structure of the information processing system.
It is composed of large number of highly interconnected simple processing elements (neurons)

Introduction to Artificial Intelligence module | PDF page 74 | lines 15-17

B. Multilayer feedforward network – The concept is of eedforward ANN having more than one weighted layer. As this network has one or more layers between the input and the output layer, it
is called hidden layers.

Question sources

Computer Science MCQ practice collection · Question 632

Question 150 | AM0633

What is the primary purpose of expert systems in artificial intelligence?

A. Emulate human expertise in a specific domain

B. Perform complex mathematical calculations

C. Process and analyze large datasets

D. Optimize computer hardware performance

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer A: Emulate human expertise in a specific domain

Expert systems solve complex problems in a particular domain at the level of human expertise. Evidence: PDF 60.

Why each choice is right or wrong

A - Correct: Exact domain-expertise purpose. See PDF p. 60.

B - Incorrect: May be a tool, not defining purpose. See PDF p. 60.

C - Incorrect: May be a workload, not defining purpose. See PDF p. 60.

D - Incorrect: Not the defining expert-system purpose. See PDF p. 60.

Primary note proof

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 60 | lines 16-18

> The expert systems are the computer applications developed to solve complex problems in a particular
> domain, at the level of extra-ordinary human intelligence and expertise. Some ccharacteristics of expert system are High performance, Understandable, Reliable, and Highly responsive.

Question sources

Computer Science MCQ practice collection · Question 633

Question 151 | AM0634

Within the cognitive approach to AI, which listed objective expresses the use of computational models to investigate human intelligence and perception?

A. Replicate human intelligence and perception

B. Enhance computer processing power

C. Analyze large datasets efficiently

D. Improve natural language processing

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer A: Replicate human intelligence and perception

Cognitive AI uses computational models to investigate biological intelligence, including human perception and thought. Replicating relevant aspects of these processes expresses that modeling objective, provided the model is compared with observations. Greater processing power and data efficiency are resources or methods. Language processing is a narrower task and need not aim to explain biological intelligence.

Why each choice is right or wrong

A - Correct: Modeling or replicating aspects of intelligence and perception is the cognitive investigation objective.

B - Incorrect: Faster hardware enables computation but does not itself supply a model of intelligence.

C - Incorrect: Efficient dataset analysis is a method or workload, not the specific cognitive-modeling objective.

D - Incorrect: NLP is one application area and does not by itself express the broader intelligence-and-perception investigation.

Primary note proof

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI02

> Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic. Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives. Thought processes and observed behavior are different evaluation targets.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 8 | lines 10-12

- Replicate human intelligent
- Solve knowledge intensive task
- Make an intelligent connection between perception and action

Question sources

Computer Science MCQ practice collection · Question 634

Question 152 | AM0635

Which AI approach uses computational models to investigate human thought processes and compares those models with human behavior?

A. Machine learning,

B. Natural language processing

C. Cognitive AI (cognitive modeling)

D. Robotics

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: Cognitive AI (cognitive modeling)

Cognitive AI investigates biological intelligence through computational models, testing those models against human behavior or experimental observations. This is a scientific modeling purpose rather than simply useful automation. Machine learning, NLP and robotics may contribute tools to cognitive research, but none is defined by that purpose. The option now uses the explicit approach name taught in the note.

Why each choice is right or wrong

A - Incorrect: Machine learning improves performance from experience and need not model human thought processes.

B - Incorrect: NLP focuses on language interpretation and generation rather than necessarily explaining human cognition.

C - Correct: Cognitive AI uses computational models to investigate biological intelligence and human thought.

D - Incorrect: Robotics integrates physical sensing and action; it need not aim to explain human thought.

Primary note proof

Study-note section AI01

Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI02

Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic. Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives. Thought processes and observed behavior are different evaluation targets.

Study-note section AI37

NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI39

Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Study-note section AI08

Learning is improvement on task T, measured by P, through experience E.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 5 | lines 12-14

the human thought process. This view is the cognitive science approach to AI. -Think human like
2. The second approach is best embodied by the concept of the Turing Test. Turing held that in future
computers can be programmed to acquire abilities rivaling human intelligence.As part of his

Introduction to Artificial Intelligence module | PDF page 7 | lines 9-10

Cognitive AI: computers are used to test theories about how the human mind works--for example,
theories about how we recognize faces and other objects, or about how we solve abstract problems.

Question sources

Computer Science MCQ practice collection · Question 635

Question 153 | AM0636

Which area of AI research focuses on understanding and replicating human vision?

A. Natural language processing

B. Computer vision

C. Robotics

D. Expert systems

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer B: Computer vision

Computer vision extracts and comprehends image information, making it the specific vision field. Evidence: PDF 84.

Why each choice is right or wrong

A - Incorrect: Language rather than image interpretation. See PDF p. 84.

B - Correct: Exact visual-comprehension field. See PDF p. 84.

C - Incorrect: Robots may use vision but robotics is broader. See PDF p. 84.

D - Incorrect: Domain expertise rather than vision specifically. See PDF p. 84.

Primary note proof

Study-note section AI39

> Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 84 | lines 6-8

accomplish automatic visual comprehension.
o Hardware of Computer Vision System - This involves –
• Power supply

Question sources

Computer Science MCQ practice collection · Question 636

Question 154 | AM0637

Which AI technique is used to extract insights and patterns from large datasets?

A. Natural language processing

B. Machine learning

C. Expert systems

D. Robotics

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Machine learning

Machine learning applications include data mining/knowledge discovery and learning patterns; ANN example p71 specifically extracts complex patterns and trends. Evidence: PDF 67, 71.

Why each choice is right or wrong

A - Incorrect: Language-specific tools do not define general data learning. See PDF p. 67, 71.

B - Correct: Matches learning/discovery from data. See PDF p. 67, 71.

C - Incorrect: Knowledge-based domain expertise rather than general learned patterns. See PDF p. 67, 71.

D - Incorrect: Physical agents rather than the broad data-analysis technique. See PDF p. 67, 71.

Primary note proof

Study-note section AI08

Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Study-note section AI31

Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI40

An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 67 | lines 29-31

• Data mining and knowledge discovery
• Speech/image/video (pattern) recognition
• Adaptive control

Introduction to Artificial Intelligence module | PDF page 71 | lines 16-18

can be used to extract patterns and detect trends that are too complex to be noticed by either humans or
other computer techniques. A trained neural network can be thought of as an "expert" in the category
of information it has been given to analyse. This expert can then be used to provide projections given

Question sources

Computer Science MCQ practice collection · Question 637

Question 155 | AM0638

Which concept in AI refers to the ability of machines to understand and interpret human language?

A. Natural language processing

B. Machine learning

C. Expert systems

D. Robotics

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Natural language processing

NLP is expressly machine understanding, analysis and interpretation of human languages. Evidence: PDF 79.

Why each choice is right or wrong

A - Correct: Exact language-understanding concept. See PDF p. 79.

B - Incorrect: Can support NLP but is broader learning from data. See PDF p. 79.

C - Incorrect: Domain expertise is not language understanding generally. See PDF p. 79.

D - Incorrect: Physical embodied action is a different focus. See PDF p. 79.

Primary note proof

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 79 | lines 5-7

> manipulate, and interpret human's languages. It helps developers to organize knowledge for performing tasks such as translation, automatic summarization, Named Entity Recognition (NER),
> speech recognition, relationship extraction, and topic segmentation.

Question sources

Computer Science MCQ practice collection · Question 638

Question 156 | AM0639

Which AI technique is used to simulate human-like conversation and interaction?

A. Machine learning

B. Natural language processing

C. Robotics

D. Expert systems

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Natural language processing

The module explicitly identifies chatbots providing customer chat as an NLP application. Evidence: PDF 82.

Why each choice is right or wrong

A - Incorrect: May power a chatbot, but not the specific application field. See PDF p. 82.

B - Correct: Explicit conversation application field. See PDF p. 82.

C - Incorrect: Embodiment not required for conversation. See PDF p. 82.

D - Incorrect: Expert advice is not the defining conversation technique. See PDF p. 82.

Primary note proof

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Study-note section AI39

Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 82 | lines 5-7

- Chatbot - is one of the important applications of NLP. It is used by many companies to provide the customer's chat services.
- Information extraction - Information extraction is one of the most important applications of

Question sources

Computer Science MCQ practice collection · Question 639

Question 157 | AM0640

What is the specifically computational-modeling contribution of cognitive AI to studying human thought, as distinct from collecting measurements or running psychological experiments alone?

A. By analyzing human brain activity

B. By replicating human thought processes

C. By conducting psychological experiments

D. By improving computer hardware capabilities

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: By replicating human thought processes

Cognitive AI contributes computational models that implement proposed mechanisms of human thought. Their predictions can then be compared with behavior, response times, errors or other observations. Brain measurements and psychological experiments can provide valuable evidence, but collecting evidence alone is distinct from constructing the computational model. Faster hardware similarly enables work without supplying a cognitive explanation.

Why each choice is right or wrong

A - Incorrect: Brain measurements can provide evidence, but measurement alone does not construct the computational thought model requested.

B - Correct: Replicating aspects of thought in a computational model provides a theory whose behavior can be tested.

C - Incorrect: Psychological experiments can test a model, but experiments alone do not supply the computational representation asked for.

D - Incorrect: More capable hardware supplies resources without itself representing a theory of human thought.

Primary note proof

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI02

> Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic. Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives. Thought processes and observed behavior are different evaluation targets.

Study-note section AI02

> The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 7 | lines 9-11

> Cognitive AI: computers are used to test theories about how the human mind works--for example,
> theories about how we recognize faces and other objects, or about how we solve abstract problems.
> 1.6Limits of AI Today

Introduction to Artificial Intelligence module | PDF page 5 | lines 11-12

> This view involves trying to understand human thought and an effort to build machines that emulate
> the human thought process. This view is the cognitive science approach to AI. -Think human like

Question sources

Computer Science MCQ practice collection · Question 640

Question 158 | AM0641

Which field builds integrated physical agents that sense their surroundings and act autonomously through motors or other actuators?

A. Machine learning

B. Robotics

C. Natural language processing

D. Expert systems

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Robotics

Robotics builds physical systems that sense, plan and act, combining mechanical, electrical, control and computing components. The requirement for integrated physical sensing and actuation distinguishes it from software autonomy in general. Machine learning, NLP and expert systems can be components of a robot, but none by itself names the field responsible for the complete physical agent.

Why each choice is right or wrong

A - Incorrect: Machine learning provides experience-based improvement but does not itself define construction of physical sensing-and-acting systems.

B - Correct: Robotics integrates the mechanical, electrical, control and computing systems of physical agents.

C - Incorrect: NLP processes human language and does not require a physical body.

D - Incorrect: Expert systems supply specialized knowledge-based reasoning and need not control physical actuators.

Primary note proof

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Study-note section AI37

NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI40

An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 82 | lines 10-12

Robotics is a domain in artificial intelligence that deals with the study of creating intelligent and efficient robots. Robots are the artificial agents acting in real world environment. Robots are aimed at
manipulating the objects by perceiving, picking, moving, modifying the physical properties of object,

Question sources

Computer Science MCQ practice collection · Question 641

Question 159 | AM0642

When studying how people recognize objects, what is the computational-modeling contribution of cognitive AI, rather than building a sensor or gathering biological and behavioral measurements alone?

A. By simulating human sensory organs

B. By analyzing human brain waves

C. By replicating human perceptual processes

D. By conducting psychological experiments

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: By replicating human perceptual processes

Cognitive AI can implement proposed perceptual processes and compare the resulting recognition behavior with human observations. This is distinct from only building an input sensor or collecting brain-wave and behavioral measurements. Those activities can inform or test the model, but they do not alone provide its computational account of perception. The scope is modeling the recognition process, not claiming that other research methods lack value.

Why each choice is right or wrong

A - Incorrect: A sensory-organ simulation models an input mechanism, but does not by itself explain how observed patterns are recognized.

B - Incorrect: Brain-wave measurements provide data rather than the computational recognition model requested.

C - Correct: Replicating aspects of perceptual processing supplies a model whose recognition behavior can be tested.

D - Incorrect: Psychological experiments provide tests or observations; by themselves they do not construct the computational perception model.

Primary note proof

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI02

> The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Study-note section AI39

> Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 7 | lines 10-12

> theories about how we recognize faces and other objects, or about how we solve abstract problems.
> 1.6Limits of AI Today
> Today‘s successful AI systems operate in well-defined domains and employ narrow, specialized

Question sources

Computer Science MCQ practice collection · Question 642

Question 160 | AM0643

Which AI technique solves specialized domain problems by applying expert knowledge encoded in a knowledge base through an inference engine?

A. Machine learning

B. Expert systems

C. Natural language processing

D. Robotics

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Expert systems

An expert system applies specialized domain knowledge through an inference engine operating on a knowledge base and working facts. This is more specific than merely exhibiting intelligent problem solving. Machine learning emphasizes improvement from experience, NLP concerns language, and robotics combines sensing, planning and physical action. Those methods may be combined with expert systems, but they are not the technique defined by the stated knowledge-based architecture.

Why each choice is right or wrong

A - Incorrect: Machine learning fits behavior from experience rather than defining this expert-knowledge architecture.

B - Correct: Expert systems use a knowledge base and inference engine for specialized domain reasoning.

C - Incorrect: NLP concerns human language rather than this domain-expertise architecture.

D - Incorrect: Robotics concerns physical sensing and action rather than the stated knowledge-base technique.

Primary note proof

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Study-note section AI37

NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI39

Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 60 | lines 17-19

domain, at the level of extra-ordinary human intelligence and expertise. Some ccharacteristics of expert system are High performance, Understandable, Reliable, and Highly responsive. Capabilities of Expert Systems include

Introduction to Artificial Intelligence module | PDF page 67 | lines 19-21

systems that learn from experience and examples. It is the computational study of algorithms that
improve performance based on experience
Machine learning is particularly attractive in several real- l i f e problems because ofthe following

Question sources

Computer Science MCQ practice collection · Question 643

Question 161 | AM0644

What is a heuristic in the context of artificial intelligence (AI)?

A. A predefined set of rules for decision making

B. A mathematical model used to solve search problems

C. An intelligent agent capable of learning from experience

D. A strategy or rule of thumb used to guide problem-solving

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer D: A strategy or rule of thumb used to guide problem-solving

A heuristic guides search toward promising states using estimated closeness; it is a problem-solving guide rather than a complete learning agent or guaranteed exact solver. Evidence: PDF 24, 24.

Why each choice is right or wrong

A - Incorrect: A predefined rule could implement a heuristic, but not all decision rules are heuristics. See PDF p. 24, 24.

B - Incorrect: A heuristic need not be a complete mathematical search model. See PDF p. 24, 24.

C - Incorrect: A learning agent is an entity, not the guiding strategy. See PDF p. 24, 24.

D - Correct: Closest to the module's guiding-way definition. See PDF p. 24, 24.

Primary note proof

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 24 | lines 17-19

> • Informed search is also called a Heuristic search. A heuristic is a way which might not always be
> guaranteed for best solutions but guaranteed to find a good solution in reasonable time.
> • Informed search can solve much complex problem which could not be solved in another way.

Introduction to Artificial Intelligence module | PDF page 24 | lines 22-24

> • It takes the current state of the agent as its input and produces the estimation of how close agent is
> from the goal.
> • The heuristic method, however, might not always give the best solution, but it guaranteed to find a

Question sources

Computer Science MCQ practice collection · Question 644

Question 162 | AM0645

How are heuristics useful in search problems?

A. They guarantee finding the optimal solution.

B. They reduce the search space and speed up the search process.

C. They eliminate the need for search algorithms.

D. They ensure a systematic exploration of the search space.

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: They reduce the search space and speed up the search process.

Problem knowledge guides search toward promising states, often reducing explored nodes and time. It does not universally guarantee optimality or replace a search algorithm. Evidence: PDF 24.

Why each choice is right or wrong

A - Incorrect: Module explicitly warns heuristic may not give best solution. See PDF p. 24.

B - Correct: Matches guided efficient exploration. See PDF p. 24.

C - Incorrect: A heuristic is used within informed search. See PDF p. 24.

D - Incorrect: Systematic exploration alone does not distinguish heuristic guidance. See PDF p. 24.

Primary note proof

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI18

> Admissibility suffices for optimal A* tree search; graph search that never reopens CLOSED states normally requires consistency. With an admissible inconsistent heuristic, reopen states when g improves. Test a goal when removed at minimum f. Completeness also needs finite branching and positive lower-bounded costs in infinite spaces. Worst-case time and memory can be exponential; no heuristic universally guarantees speed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 24 | lines 16-18

> • Informed search strategies can find a solution more efficiently than an uninformed search strategy.

• Informed search is also called a Heuristic search. A heuristic is a way which might not always be
guaranteed for best solutions but guaranteed to find a good solution in reasonable time.

Question sources

Computer Science MCQ practice collection · Question 645

Question 163 | AM0646

Which general characterization says that search uses problem-specific knowledge to estimate remaining cost, without itself requiring either non-overestimation or an edge-by-edge consistency inequality?

A. Admissible heuristic

B. Consistent heuristic

C. Informed heuristic

D. Dominance comparison between heuristics

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Informed heuristic

Informed search adds problem-specific knowledge to guide exploration, often through a remaining-cost estimate. That general characterization does not guarantee admissibility or consistency, which impose additional mathematical conditions. Dominance instead compares two estimates pointwise. The question asks for the broad knowledge-use property rather than one of those stronger conditions or comparisons.

Why each choice is right or wrong

A - Incorrect: Admissibility adds a non-overestimation bound, stronger than the general knowledge-use characterization requested.

B - Incorrect: Consistency adds an edge-by-edge inequality, stronger than the general property requested.

C - Correct: Informed heuristic guidance uses problem-specific knowledge to estimate remaining cost or promise.

D - Incorrect: Dominance compares the numerical estimates of two heuristics rather than naming general use of problem knowledge.

Primary note proof

Study-note section AI17

A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI18

A* orders by f(n)=g(n)+h(n). Admissibility means $0 \le h(n) \le h^*(n)$, the true cheapest remaining cost. Consistency means $h(n) \le c(n,n') + h(n')$ on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI-S05

For two admissible heuristics, h_2 dominates h_1 when $h_2(n) \ge h_1(n)$ for every node. Dominance compares estimates; it is not itself a condition ensuring that an arbitrary heuristic never overestimates. Both estimates must separately satisfy the relevant admissibility assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 24 | lines 14-16

• It uses domain knowledge. In an informed search, problem information is available which can guide
the search.
• Informed search strategies can find a solution more efficiently than an uninformed search strategy.

Question sources

Computer Science MCQ practice collection · Question 646

Question 164 | AM0647

Which term names the heuristic property defined by $0 \le h(n) \le h^*(n)$, where $h^*(n)$ is the true cheapest remaining cost?

A. Admissible heuristic

B. Consistent heuristic

C. Informed heuristic

D. Dominance comparison between heuristics

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: Admissible heuristic

The displayed non-overestimation inequality defines admissibility. Consistency is a different edge condition; with the usual nonnegative heuristic estimates and a zero goal estimate, it implies admissibility for states with a goal path. Thus a consistent heuristic can also satisfy the bound, but consistency is not the name of the property being defined. Informedness and dominance do not independently establish the bound.

Why each choice is right or wrong

A - Correct: Admissibility is exactly the non-overestimation property stated by this inequality.

B - Incorrect: Consistency is an edge-by-edge inequality with goal normalization, rather than the name of the displayed non-overestimation bound.

C - Incorrect: Being informed only means using problem knowledge and does not impose this bound.

D - Incorrect: Dominance compares two heuristic estimates and does not by itself bound either one by the true remaining cost.

Primary note proof

Study-note section AI18

> A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI-S05

> For two admissible heuristics, h_2 dominates h_1 when $h_2(n) \geq h_1(n)$ for every node. Dominance compares estimates; it is not itself a condition ensuring that an arbitrary heuristic never overestimates. Both estimates must separately satisfy the relevant admissibility assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 25 | lines 4-6

> • Admissibility of the heuristic function is given as: h(n) <= h*(n) Here h(n) is heuristic cost, and h*(n) is the estimated cost. Hence heuristic cost should be less than or equal to the estimated cost.
> • Pure Heuristic Search - Pure heuristic search is the simplest form of heuristic search algorithms. It

Introduction to Artificial Intelligence module | PDF page 28 | lines 17-18

> o Consistency: Second required condition is consistency for only A* graph-search.
> • If the heuristic function is admissible, then A* tree search will always find the least cost path.

Question sources

Computer Science MCQ practice collection · Question 647

Question 165 | AM0648

Which forward best-first algorithm ranks frontier nodes by accumulated path cost plus a heuristic estimate of remaining cost?

A. Depth-first search

B. Breadth-first search

C. A* search algorithm

D. Hill climbing algorithm

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: A* search algorithm

A* orders frontier nodes by f(n) = g(n) + h(n), combining the cost already paid with an estimate of the cost still needed. Estimating future cost is not a backward search frontier. DFS and BFS use depth-related removal orders, while hill climbing improves neighboring complete candidates. Replacing the mistaken backward-search phrase preserves the intended heuristic-and-cost algorithm identification.

Why each choice is right or wrong

A - Incorrect: DFS follows a stack order and does not inherently combine a heuristic with backward search.

B - Incorrect: BFS uses breadth order without requiring heuristics or a backward frontier.

C - Correct: A* is forward best-first search using f = g + h.

D - Incorrect: Hill climbing selects improving neighbors rather than ordering a frontier by accumulated plus remaining path cost.

Primary note proof

Study-note section AI18

> A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI13

> BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Study-note section AI20

Local search keeps a complete candidate configuration and improves its objective; the path may be irrelevant. Hill climbing chooses better neighbors. First-improvement accepts the first improvement; steepest-ascent examines all neighbors; stochastic variants choose among improvements. Local maxima, plateaus and ridges can stop progress. Random restarts or limited sideways steps can help without guaranteeing success.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 26 | lines 25-27

• A* search is the most commonly known form of best-first search.
• It uses heuristic function h(n), and cost to reach the node n from the start state g(n). It has combined
features of UCS and greedy best-first search, by which it solves the problem efficiently.

Introduction to Artificial Intelligence module | PDF page 27 | lines 20-25

o Place the starting node in the OPEN list.
o Check if the OPEN list is empty or not, if the list is empty then return failure and stops.
o Select the node from the OPEN list which has the smallest value of evaluation function (g+h), if node n is goal node then return success and stop, otherwise
o Expand node n and generate all of its successors, and put n into the closed list. For each successor n', check whether n' is already in the OPEN or CLOSED list, if not then compute

Introduction to Artificial Intelligence module | PDF page 23 | lines 17-19

forward-search and other from goal node called as backward-search, to find the goal node.
• Bidirectional search replaces one single search graph with two small sub-graphs in which one starts
the search from an initial vertex and other starts from goal vertex.

Question sources

Computer Science MCQ practice collection · Question 648

Question 166 | AM0649

What is the main advantage of using heuristics in game playing AI systems?

A. They guarantee victory in every game.

B. They make the game unpredictable and exciting.

C. They allow the AI to learn from its opponents.

D. They improve the AI's decision-making and gameplay.

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer D: They improve the AI's decision-making and gameplay.

Game search computes candidate moves with a heuristic and chooses the best backed-up outcome, supporting improved decision quality rather than automatic victory or learning. Evidence: PDF 33, 34.

Why each choice is right or wrong

A - Incorrect: Optimal opponent may prevent a win. See PDF p. 33, 34.

B - Incorrect: Unpredictability is not the stated purpose. See PDF p. 33, 34.

C - Incorrect: Heuristic evaluation is not necessarily learning. See PDF p. 33, 34.

D - Correct: Matches heuristic move evaluation and selection. See PDF p. 33, 34.

Primary note proof

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Study-note section AI22

> Adversarial search models agents with competing objectives. A game specifies states, player to move, legal actions, successor function, terminal test and utility. Perfect information means previous moves and current state are visible; it does not reveal future choices or imply absence of chance. Zero-sum utility makes one player’s gain the other’s loss.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 33 | lines 35-35

> • Compute the move using a heuristic function.

Introduction to Artificial Intelligence module | PDF page 34 | lines 3-4

> • Make the best move from the choices.
> • For example, in the above figure, the two players MAX and MIN are there. MAX starts the game

Question sources

Computer Science MCQ practice collection · Question 649

Question 167 | AM0650

Which component is the scoring mapping that assigns a heuristic desirability value to a game state, rather than a search, pruning or sampling procedure?

A. Minimax algorithm

B. Alpha-beta pruning

C. Monte Carlo simulation

D. Evaluation function

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer D: Evaluation function

An evaluation function assigns a desirability score to a game state, especially when search stops at a depth cutoff. Minimax uses scores to choose moves, and alpha-beta avoids branches that cannot alter the backed-up result. Monte Carlo sampling can estimate values from possible continuations, but the sampling procedure is distinct from the mapping whose output is a state score.

Why each choice is right or wrong

A - Incorrect: Minimax backs values up through competing choices; it uses state scores rather than being the score mapping itself.

B - Incorrect: Alpha-beta pruning skips branches that cannot affect a minimax decision; it is not the scoring mapping.

C - Incorrect: Monte Carlo simulation samples continuations whose outcomes may estimate value; sampling is a procedure, not the score mapping itself.

D - Correct: An evaluation function maps a state to its estimated desirability score.

Primary note proof

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Study-note section AI23

> Alpha is MAX’s best lower bound; beta is MIN’s best upper bound. Pass bounds down, update alpha at MAX and beta at MIN, and prune when alpha≥beta. Pruning preserves the minimax value for the same searched tree. Good move ordering increases pruning: worst-case O(b^m), ideal ordering about O(b^(m/2)).

Study-note section AI22

Adversarial search models agents with competing objectives. A game specifies states, player to move, legal actions, successor function, terminal test and utility. Perfect information means previous moves and current state are visible; it does not reveal future choices or imply absence of chance. Zero-sum utility makes one player's gain the other's loss.

Study-note section AI-S05

A Monte Carlo game simulation samples possible continuations and uses their outcomes to estimate position value. An evaluation function is the scoring mapping itself; sampling is one way of obtaining an estimate, while minimax backs values up and alpha-beta removes branches that cannot affect that backed-up decision.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 33 | lines 6-8

• Games are modeled as a Search problem and heuristic evaluation function, and these are the two
main factors which help to model and solve games in AI.
1. Minimax algorithm

Question sources

Computer Science MCQ practice collection · Question 650

Question 168 | AM0651

What is the purpose of using heuristics in game tree search algorithms?

A. To guarantee a win in every game.

B. To eliminate the need for search algorithms.

C. To reduce the search space and improve efficiency.

D. To ensure a systematic exploration of the search tree.

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: To reduce the search space and improve efficiency.

Pruning removes nodes that cannot affect the decision and speeds game search; heuristic domain-informed ordering improves pruning effectiveness. This supports efficiency, not guaranteed wins or eliminating search. Evidence: PDF 35, 35.

Why each choice is right or wrong

A - Incorrect: A better search can still lose to stronger positions. See PDF p. 35, 35.

B - Incorrect: Pruning/search guidance still requires search. See PDF p. 35, 35.

C - Correct: Matches reducing irrelevant exploration. See PDF p. 35, 35.

D - Incorrect: Systematic traversal does not describe the efficiency purpose. See PDF p. 35, 35.

Primary note proof

Study-note section AI23

> Alpha is MAX's best lower bound; beta is MIN's best upper bound. Pass bounds down, update alpha at MAX and beta at MIN, and prune when alpha≥beta. Pruning preserves the minimax value for the same searched tree. Good move ordering increases pruning: worst-case O(b^m), ideal ordering about O(b^(m/2)).

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 35 | lines 8-10

> • It removes all the nodes which are not really affecting the final decision but making algorithm slow.
> Hence by pruning these nodes, it makes the algorithm fast.
> • The main condition which required for alpha-beta pruning is α>=β

Introduction to Artificial Intelligence module | PDF page 35 | lines 32-33

> • Use domain knowledge while finding the best move.
> • We can bookkeep the states, as there is a possibility that states may repeat.

Question sources

Computer Science MCQ practice collection · Question 651

Question 169 | AM0652

Which game-state heuristic evaluates the weighted number and value of pieces held by each player, rather than legal move count or controlled territory?

A. Material heuristic

B. Mobility heuristic

C. Distance heuristic

D. Tactical heuristic

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer A: Material heuristic

Material evaluation counts pieces with weights reflecting their values. Mobility instead measures legal options, while territory evaluates controlled regions. Restricting the stem to weighted pieces removes the original conflation of material and territory. Distance and tactical features may inform other evaluations, but they do not name the weighted piece-count feature requested. The distinction concerns what the evaluator measures.

Why each choice is right or wrong

A - Correct: Material is the weighted piece-count/value feature.

B - Incorrect: Mobility measures available legal moves rather than the amount of material or territory controlled.

C - Incorrect: Distance evaluates spatial separation, not the piece-count or controlled-region feature stated here.

D - Incorrect: This does not name weighted piece holdings; the feature described is material evaluation.

Primary note proof

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 33 | lines 6-8

> • Games are modeled as a Search problem and heuristic evaluation function, and these are the two
> main factors which help to model and solve games in AI.
> 1. Minimax algorithm

Question sources

Computer Science MCQ practice collection · Question 652

Question 170 | AM0653

Which heuristic evaluates a game state by considering the available moves and options for each player?

A. Material heuristic

B. Mobility heuristic

C. Distance heuristic

D. Tactical heuristic

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: The mobility heuristic.

AI23 explicitly says mobility counts legal options. Available moves and options for each player are exactly that feature, making mobility the supported answer. Material measures weighted pieces, while distance and tactics refer to different aspects of a position. The heuristic may be combined with other features, but the specific feature requested is unambiguous.

Why each choice is right or wrong

A - Incorrect: Material evaluates the amount or value of pieces, not the number of available moves.

B - Correct: Mobility directly counts or scores the legal move options available to a player.

C - Incorrect: Distance concerns separation or proximity, not the legal-option count described in the question.

D - Incorrect: Tactical evaluation concerns concrete threats or sequences rather than the specific mobility feature requested.

Primary note proof

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 33 | lines 6-8

> • Games are modeled as a Search problem and heuristic evaluation function, and these are the two

main factors which help to model and solve games in AI.
1. Minimax algorithm

Question sources

Computer Science MCQ practice collection · Question 653

Question 171 | CS0200

_____ is a set of procedures that uses representational language to conclude new facts from existing ones.

A. Inference Engine

B. Explanatory facility

C. Interface

D. Knowledge Base

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer A: Inference Engine

Inference engine manipulates knowledge and repeatedly applies rules to facts to derive new knowledge. The knowledge base stores facts, while the user interface handles interaction/explanation. Evidence: PDF 63, 63, 64.

Why each choice is right or wrong

A - Correct: Matches procedural derivation of new facts. See PDF p. 63, 63, 64.

B - Incorrect: Explaining an outcome differs from deriving facts. See PDF p. 63, 63, 64.

C - Incorrect: Interaction layer differs from inference procedures. See PDF p. 63, 63, 64.

D - Incorrect: Stores knowledge rather than the inference procedure. See PDF p. 63, 63, 64.

Primary note proof

Study-note section AI40

An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Study-note section AI24

Knowledge representation encodes facts, objects, relations, events and rules so a machine can use them. Syntax defines legal expressions; semantics defines their meaning. A knowledge base

stores assertions; an inference system derives consequences. Declarative knowledge says what is true; procedural knowledge specifies how to act.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 63 | lines 17-19

• In case of knowledge-based ES, the Inference Engine acquires and manipulates the knowledge from
the knowledge base to arrive at a particular solution.
• In case of rule-based ES, it –

Introduction to Artificial Intelligence module | PDF page 63 | lines 20-22

o Applies rules repeatedly to the facts, which are obtained from earlier rule application.
o Adds new knowledge into the knowledge base if required.
o Resolves rules conflict when multiple rules are applicable to a particular case

Introduction to Artificial Intelligence module | PDF page 64 | lines 20-22

• User interface provides interaction between user of the ES and the ES itself.
• Natural Language Processing so as to be used by the user who is well-versed in the task domain
• It explains how the ES has arrived at a particular recommendation.

Question sources

Computer Science Exit Exam practice collection · Question 200

Question 172 | CS0201

Which one is a space where an agent perceives and acts, so that an intelligent agent designer has to understand the type of it?

A. Environment

B. Actuator

C. Sensor

D. Search space

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer A: Environment

Environment is the surrounding space in which an agent operates and senses/acts; sensors and actuators are interfaces to it. Evidence: PDF 13.

Why each choice is right or wrong

A - Correct: Exact surrounding operating context. See PDF p. 13.

B - Incorrect: Action-output component. See PDF p. 13.

C - Incorrect: Perceptual-input component. See PDF p. 13.

D - Incorrect: Abstract candidate-solution set, not necessarily the operating surroundings. See PDF p. 13.

Primary note proof

Study-note section AI03

> An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Study-note section AI03

> A percept is one observation; a percept sequence is the entire observation history. An agent function maps percept sequences to actions: f:P*→A. An agent program implements that mapping on an architecture; agent = architecture + program. Environment means the setting in which it senses and acts, not merely the search space. Autonomy means relying substantially on experience rather than only a designer's prior instructions.

Study-note section AI12

> A search node records a state plus parent, action, depth and path cost. Expanding a node generates its successors. OPEN or frontier stores discovered candidates awaiting expansion; CLOSED records expanded states. Reaching the same state through different paths does not imply the same node or cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 13 | lines 4-6

> • The environment is where agent lives, operate and provide the agent with something to sense and
> act upon it.
> • The agent takes input from the environment through sensors and delivers the output to the

Question sources

Computer Science Exit Exam practice collection · Question 201

Question 173 | CS0202

Which one of the following is an uninformed searching method?

A. A* search

B. Iterative improvement

C. Breadth first search

D. Greedy search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Breadth first search

The listed uninformed strategies include BFS. A* and best-first are separately listed informed strategies; local iterative improvement is not the cited uninformed breadthwise procedure. Evidence: PDF 17, 17.

Why each choice is right or wrong

A - Incorrect: Informed heuristic method. See PDF p. 17, 17.

B - Incorrect: Local improvement is not the listed uninformed method. See PDF p. 17, 17.

C - Correct: Explicit uninformed method. See PDF p. 17, 17.

D - Incorrect: Informed heuristic best-first method. See PDF p. 17, 17.

Primary note proof

Study-note section AI13

> BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI20

> Local search keeps a complete candidate configuration and improves its objective; the path may be irrelevant. Hill climbing chooses better neighbors. First-improvement accepts the first improvement; steepest-ascent examines all neighbors; stochastic variants choose among improvements. Local maxima, plateaus and ridges can stop progress. Random restarts or limited sideways steps can help without guaranteeing success.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 17 | lines 35-37

> o Unformed Search Strategies
> - DFS, BFS, Uniform cost search, DLS, Iterative Deeping DFS, and Bi-directional search.

Introduction to Artificial Intelligence module | PDF page 17 | lines 38-41

o Informed Search Strategies
- A star search, Best first search
o Local Search Strategies
- Hill climbing search, Means ends analysis, Simulated annealing

Question sources

Computer Science Exit Exam practice collection · Question 202

Question 174 | CS0203

Which one of the following searching strategies best explains: 'The searching agent is based on problem-specific knowledge and the agent knows how far away a different state is from the goal'?

A. Blind search

B. Informed search

C. Uninformed search

D. Not informed search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Informed search

Problem-specific knowledge of estimated distance to the goal is the module's informed/heuristic-search distinction. Blind search lacks such guidance. Evidence: PDF 24, 18.

Why each choice is right or wrong

A - Incorrect: Blind search omits goal-closeness knowledge. See PDF p. 24, 18.

B - Correct: Exact informed-search distinction. See PDF p. 24, 18.

C - Incorrect: Lacks the stated domain guidance. See PDF p. 24, 18.

D - Incorrect: Synonymous with uninformed/blind search. See PDF p. 24, 18.

Primary note proof

Study-note section AI17

A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Introduction to Artificial Intelligence module | PDF page 24 | lines 14-16

• It uses domain knowledge. In an informed search, problem information is available which can guide
the search.
• Informed search strategies can find a solution more efficiently than an uninformed search strategy.

Introduction to Artificial Intelligence module | PDF page 18 | lines 23-25

• The uninformed search does not contain any domain knowledge such as closeness, the location of
the goal. It is also called level order traversal.
• It operates in a brute-force way as it only includes information about how to traverse the tree and

Question sources

Computer Science Exit Exam practice collection · Question 203

Question 175 | NS0001

Which type of AI is capable of performing any intellectual task that a human can do?

A. Narrow AI

B. General AI

C. Machine Learning

D. Expert Systems

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: General AI.

AI01 defines general AI by broad competence transferable across domains, matching the hypothetical ability to perform any human intellectual task. Narrow AI is bounded; machine learning names a family of methods and expert systems supply specialized domain knowledge. Neither automatically gives universal intellectual capability. The statement describes a category, not a claim that such a system presently exists.

Why each choice is right or wrong

A - Incorrect: Narrow AI is limited to bounded tasks rather than broad intellectual transfer.

B - Correct: General AI is the broad cross-domain competence category described.

C - Incorrect: Machine learning is a method family and does not itself imply general intellectual competence.

D - Incorrect: Expert systems apply specialized domain knowledge, not arbitrary human intellectual capability.

Primary note proof

Study-note section AI01

> Narrow or weak AI addresses bounded tasks, such as face recognition, spam filtering or a voice assistant. General AI denotes broad competence transferable across domains; strong AI also has a philosophical meaning involving actual understanding. Do not equate a humanoid body, autonomous vehicle, or many features with general intelligence.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Study-note section AI40

> An expert system applies specialized knowledge to a domain such as equipment diagnosis. The domain expert supplies expertise; the knowledge engineer elicits and encodes it; the inference engine reasons over a knowledge base and working facts. The user interface handles interaction; an explanation facility presents why or how a conclusion was reached.

Question sources

Artificial Intelligence review questions · Question 1

Question 176 | NS0002

What is the primary goal of Machine Learning?

A. To hard-code rules into a machine

B. To allow systems to learn and improve from experience without being explicitly programmed

C. To simulate physical environments

D. To build robotic hardware

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Improve task performance through experience.

AI08 defines learning as improvement on a task, measured by performance, through experience. B captures that adaptive objective; without explicitly programmed means the learned behavior is not simply an exhaustive hand-coded rule list, not that no algorithm is programmed. Simulation and robot construction can use ML but do not define its primary goal.

Why each choice is right or wrong

A - Incorrect: Hard-coding all task rules is not the experience-based improvement defining machine learning.

B - Correct: The system improves measured task performance from experience rather than only fixed task rules.

C - Incorrect: Physical simulation is one application context, not the definition or primary learning objective.

D - Incorrect: Building hardware is not required for machine learning, which also operates in software.

Primary note proof

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Study-note section AI-S01

> A conventional computer manipulates digital information according to a program or sequence of instructions. Programs can implement storage, retrieval and learning operations. Neural networks can learn pattern recognition from examples, so executing instructions and learning patterns are not mutually exclusive descriptions. This provides no universal claim that every machine is better or worse than every human at completing missing information.

Question sources

Artificial Intelligence review questions · Question 2

Question 177 | NS0003

Which reasoning type guarantees that if premises are true, the conclusion is also true?

A. Inductive Reasoning

B. Abductive Reasoning

C. Deductive Reasoning

D. Probabilistic Reasoning

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer C: Deductive reasoning.

AI28 defines deduction by truth-preserving rules and contrasts induction's revisable generalizations and abduction's plausible explanations. With valid deductive inference and true premises, the conclusion must be true. Probabilistic support expresses uncertainty rather than this logical guarantee, so deductive is the relevant reasoning type.

Why each choice is right or wrong

A - Incorrect: Induction generalizes beyond observations and can be overturned by new examples.

B - Incorrect: Abduction proposes plausible explanations without guaranteeing that they are true.

C - Correct: Valid deductive rules preserve truth from true premises to the conclusion.

D - Incorrect: Probability can quantify uncertain conclusions rather than provide the stated deductive guarantee.

Primary note proof

Study-note section AI28

> Deduction applies truth-preserving rules: all people are mortal; Ada is a person; therefore Ada is mortal. Induction generalizes observations to a broader rule and can be overturned by new examples. Abduction proposes a plausible explanation: wet ground suggests rain, although sprinklers could explain it. Analogy transfers a pattern between similar situations.

Study-note section AI29

> Probability describes uncertainty about events. $P(\neg A)=1-P(A)$; $P(A|B)=P(A\cap B)/P(B)$, for $P(B)>0$. Bayes gives $P(H|E)=P(E|H)P(H)/P(E)$: posterior equals likelihood times prior divided by evidence probability. For mutually exclusive exhaustive hypotheses, $P(E)=\Sigma i P(E|Hi)P(Hi)$.

Question sources

Artificial Intelligence review questions · Question 3

Question 178 | NS0004

A self-driving car's environment is best classified as:

A. Fully Observable, Deterministic, Static

B. Partially Observable, Stochastic, Dynamic

C. Fully Observable, Stochastic, Static

D. Partially Observable, Deterministic, Dynamic

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Partially observable, stochastic and dynamic.

AI09 describes driving as partially observable, sequential and dynamic. Traffic participants and noisy observations make outcomes uncertain rather than fully determined by the car's chosen action. B matches that ordinary driving setting. Neither full visibility nor a static world fits roads whose relevant conditions change while the agent deliberates.

Why each choice is right or wrong

A - Incorrect: Driving has hidden/noisy information and a changing world rather than full observability and static conditions.

B - Correct: Hidden state, uncertain outcomes and ongoing traffic changes fit the ordinary driving task.

C - Incorrect: Full observability and static conditions do not fit the stated driving environment.

D - Incorrect: Partial observation and dynamics fit, but deterministic outcomes ignore the uncertainty in ordinary driving.

Primary note proof

Study-note section AI09

> Fully observable means sensors reveal relevant state; partially observable means relevant information is hidden or noisy. Deterministic means state and action determine the next state; stochastic means outcomes involve randomness. Strategic uncertainty about an opponent is distinct from dice-like randomness.

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Question sources

Artificial Intelligence review questions · Question 4

Question 179 | NS0005

In the A* algorithm, what does the evaluation function f(n) represent?

A. Only the heuristic cost to goal

B. Only the cost from start to node n

C. The sum of cost from start to n (g(n)) and estimated cost from n to goal (h(n))

D. The product of g(n) and h(n)

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: f(n)=g(n)+h(n).

AI18 directly defines A* evaluation as actual start-to-node cost g plus estimated node-to-goal cost h. This estimates a solution cost through the node. Using only h is greedy best-first, using only g is uniform-cost selection, and multiplication is not the A* rule.

Why each choice is right or wrong

A - Incorrect: Using h alone is greedy best-first rather than A* evaluation.

B - Incorrect: Using g alone is uniform-cost ordering and omits the remaining-cost estimate.

C - Correct: A* adds the accumulated path cost and estimated remaining cost.

D - Incorrect: The defined evaluation is a sum, not a product.

Primary note proof

Study-note section AI18

> A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI15

> UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Question sources

Artificial Intelligence review questions · Question 5

Question 180 | NS0006

Which reflex architecture maintains an internal world state and applies condition-action rules to that modeled state, rather than using only the current percept?

A. Simple Reflex Agent

B. Goal-Based Agent

C. Model-Based Reflex Agent

D. Learning Agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer C: Model-Based Reflex Agent

Model-based reflex agents update internal state from previous state, action and current observation, using models of world change and action effects. They then apply rules to that modeled state. Simple reflex agents lack persistent world-state memory. Goal-based and learning capabilities may coexist with a model, but the question specifically asks for the reflex architecture defined by state maintenance and rule application.

Why each choice is right or wrong

A - Incorrect: Simple reflex uses only the current percept without persistent world-state memory.

B - Incorrect: Goal-based choice uses future sequences reaching desired states, whereas the requested architecture applies reflex rules to modeled state.

C - Correct: Model-based reflex maintains internal state and applies rules to that state.

D - Incorrect: Learning concerns improving the agent from experience rather than naming this reflex architecture.

Primary note proof

Study-note section AI06

> A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean. It has no persistent world-state memory. This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

Study-note section AI06

> A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects. It then applies rules to that state. Both kinds can use condition-action rules; memory and a world model distinguish them.

Study-note section AI07

A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: $EU(a)=\Sigma s\ P(s|a,evidence)U(s)$. Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Study-note section AI08

The performance element selects actions. The learning element changes the agent to improve future performance. The critic evaluates behavior against a performance standard and supplies feedback. The problem generator suggests exploratory actions likely to yield useful experience.

Question sources

Artificial Intelligence review questions · Question 6

Question 181 | NS0007

What distinguishes a Rational Agent from an Omniscient Agent?

A. Rational agents always make perfect decisions

B. Rational agents act on best available information but can make mistakes; omniscient agents know all outcomes

C. Omniscient agents are a common AI type

D. Rational agents do not use information

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Rational choice uses available information; omniscience knows actual outcomes.

AI04 defines rationality by best expected performance using information available, while omniscience means knowing actual outcomes in advance. Rational actions can fail and do not require perfect predictions. The contrast is epistemic, not whether an agent uses any information or whether omniscient systems are common implementations.

Why each choice is right or wrong

A - Incorrect: Rationality does not guarantee perfect outcomes or foreknowledge.

B - Correct: This distinguishes information-conditioned expected performance from knowing actual outcomes.

C - Incorrect: The note treats omniscience as an idealized outcome-knowledge condition, not a common AI implementation.

D - Incorrect: Rationality explicitly depends on percepts, prior knowledge and available actions.

Primary note proof

Study-note section AI04

> A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Study-note section AI04

> Omniscience would mean knowing actual action outcomes in advance. Rationality does not require omniscience or guaranteed success. Even complete outcome knowledge cannot create a successful action when every available action fails. Information gathering and exploration may therefore be rational actions.

Question sources

Artificial Intelligence review questions · Question 7

Question 182 | NS0008

Which search algorithm is guaranteed to find the shortest path in an unweighted graph?

A. Depth-First Search

B. Depth-Limited Search

C. Breadth-First Search

D. Best-First Search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Breadth-first search.

AI13 says BFS explores levels in increasing depth and minimizes the number of edges. An unweighted graph gives equal step costs, making that shortest-path property applicable with visited-state tracking. DFS or a fixed depth limit does not guarantee minimum depth, and generic best-first needs its evaluation rule specified.

Why each choice is right or wrong

A - Incorrect: DFS can reach a longer path first and is not generally optimal.

B - Incorrect: A depth limit can miss a solution or return a nonshortest path within its bound.

C - Correct: BFS explores increasing edge counts and finds shortest paths in an unweighted graph.

D - Incorrect: Best-first is a family; an unspecified evaluation rule does not guarantee shortest paths.

Primary note proof

Study-note section AI13

> BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI14

> DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Study-note section AI16

> DLS runs DFS to depth l and reports cutoff separately from failure. It misses goals deeper than l. Iterative deepening runs DLS with limits 0,1,2,...; for finite b it finds the shallowest solution with O(b^d) time and O(bd) frontier space, optimal for equal step costs. Repeated shallow work is small compared with the deepest level.

Question sources

Artificial Intelligence review questions · Question 8

Question 183 | NS0009

Why is Uniform Cost Search (UCS) NOT considered a heuristic approach?

A. It doesn't consider costs

B. It uses cost as a known factor but employs no domain-specific knowledge or estimates

C. It uses heuristic functions

D. It only works on trees, not graphs

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: UCS uses known accumulated cost without a heuristic estimate.

AI15 orders UCS by actual path cost g, and AI17 explicitly calls it uninformed despite using g. Known action costs come from the problem definition; a heuristic adds an estimate of future cost. UCS can operate on graphs with duplicate handling, so neither ignoring costs nor tree-only operation explains the distinction.

Why each choice is right or wrong

A - Incorrect: UCS explicitly orders its frontier by accumulated path cost.

B - Correct: Known costs are used, but no problem-specific estimate of remaining cost is required.

C - Incorrect: UCS does not require an h estimate; g is actual accumulated cost.

D - Incorrect: UCS works on graphs with proper relaxation and duplicate handling.

Primary note proof

Study-note section AI15

> UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Question sources

Artificial Intelligence review questions · Question 9

Question 184 | NS0010

Which data structure supplies the defining next-node removal order of a depth-first search frontier?

A. Queue (FIFO)

B. Stack (LIFO)

C. Priority Queue

D. Hash Table

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Stack (LIFO)

DFS follows the deepest pending branch using a LIFO stack or recursion. That removal discipline distinguishes it from breadth-first FIFO selection and cost- or heuristic-priority selection. A hash table can still be used alongside DFS to record visited states. The question therefore specifies frontier removal rather than broadly asking which structures may track nodes.

Why each choice is right or wrong

A - Incorrect: A FIFO queue removes the oldest entry, giving breadth-first rather than depth-first frontier order.

B - Correct: A LIFO stack processes the most recently pending branch first, implementing DFS.

C - Incorrect: A priority queue removes according to a key rather than the defining DFS LIFO order.

D - Incorrect: A hash table may store visited states, but does not supply the frontier's next-node removal order.

Primary note proof

Study-note section AI14

> DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Study-note section AI13

> BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI15

UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Question sources

Artificial Intelligence review questions · Question 10

Question 185 | NS0011

What is the main limitation of Depth-First Search regarding completeness?

A. It uses too much memory

B. It can get stuck in cycles or infinite paths and fail to find a solution

C. It always finds suboptimal solutions

D. It only works for trees

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Cycles or infinite branches can prevent finding an existing solution.

AI14 says DFS can fail on infinite paths or cycles and is not generally optimal. Without suitable restrictions or duplicate handling, it may never return to a pending branch containing a goal. Finite visited-state graph traversal changes that result; the limitation is therefore not that DFS only works on trees or always returns a bad path.

Why each choice is right or wrong

A - Incorrect: DFS is usually memory-efficient; excess storage is not its defining completeness failure.

B - Correct: It may follow an infinite branch or cycle indefinitely while a solution waits elsewhere.

C - Incorrect: A returned path can be optimal by coincidence; always suboptimal is too strong.

D - Incorrect: DFS also traverses graphs, especially with visited-state tracking.

Primary note proof

Study-note section AI14

> DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Study-note section AI14

> Implicit-tree worst-case time is O(b^m), and space is O(bm) when pending siblings are stored, or O(m) for path-only recursive generation. DFS is not generally optimal and can fail on infinite paths or cycles. On a finite graph, visited-state tracking makes traversal complete but needs O(V) storage. Do not confuse the O(b^m) worst-case time bound with the DFS space bounds above.

Question sources

Artificial Intelligence review questions · Question 11

Question 186 | NS0012

Which options can be used without a training set of labeled input-target examples? Select all that apply. For D, consider whether some deep-learning objectives permit unlabeled or reward feedback.

A. Supervised Learning

B. Reinforcement Learning

C. Unsupervised Learning

D. Deep Learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B, C, D: Reinforcement Learning; Unsupervised Learning; Deep Learning

Unsupervised learning discovers structure without target labels. Reinforcement learning can learn from rewards rather than correct-action target examples. Deep learning is an architecture family and can be trained with unsupervised or reinforcement objectives, so it also need not require labeled input-target data. Supervised learning is specifically defined by those labeled pairs. The capability wording permits both regimes and architecture families to be judged accurately.

Why each choice is right or wrong

A - Incorrect: Supervised learning uses labeled examples by definition.

B - Correct: RL learns through rewards and interaction, without requiring labeled correct actions.

C - Correct: Unsupervised learning explicitly seeks structure without target labels.

D - Correct: Deep learning can use unlabeled or reward objectives, so labels are not inherently required.

Primary note proof

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

> Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI43

ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Question sources

Artificial Intelligence review questions · Question 12

Question 187 | NS0013

For A* tree search with finite branching, step costs bounded below by a positive constant, goal testing on removal and h(goal)=0, which listed heuristic condition is the weakest sufficient guarantee of optimality? Treat monotone as synonymous with consistent.

A. Consistent

B. Admissible (never overestimates the actual cost)

C. Monotone

D. Domain-independent

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Admissible (never overestimates the actual cost)

Admissibility requires that a heuristic never overestimate the true cheapest remaining cost. It suffices for optimal A* tree search under the stated cost and goal-testing assumptions. Consistency, also called monotonicity here, is stronger and is particularly important when graph search permanently closes states. Domain independence is unrelated to the optimality condition. Asking for the weakest sufficient listed condition distinguishes admissibility from the stronger alternatives.

Why each choice is right or wrong

A - Incorrect: Consistency is sufficient but stronger than the weakest listed condition for tree A*.

B - Correct: Admissibility is sufficient for optimal tree A* under the stated assumptions.

C - Incorrect: Monotonicity is consistency here and is stronger than necessary for the tree-search guarantee.

D - Incorrect: Domain independence is not a heuristic optimality condition.

Primary note proof

Study-note section AI18

> A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI18

> Admissibility suffices for optimal A* tree search; graph search that never reopens CLOSED states normally requires consistency. With an admissible inconsistent heuristic, reopen states when g improves. Test a goal when removed at minimum f. Completeness also needs finite branching and positive lower-bounded costs in infinite spaces. Worst-case time and memory can be exponential; no heuristic universally guarantees speed.

Question sources

Artificial Intelligence review questions · Question 13

Question 188 | NS0014

Iterative Deepening Search (IDS) combines the advantages of which two algorithms?

A. UCS and A*

B. BFS and DFS

C. Greedy Search and BFS

D. DLS and UCS

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Breadth-first solution depth with depth-first memory use.

AI16 defines IDS as DLS runs at limits 0,1,2,..., finding the shallowest solution under finite branching with O(bd) frontier space. It obtains BFS's shallow-solution behavior and DFS-style memory use. It does not combine UCS priorities, greedy ordering or A* cost thresholds; IDA* is a separate threshold-based variant.

Why each choice is right or wrong

A - Incorrect: IDS does not combine UCS costs with A* evaluation.

B - Correct: It obtains breadth-first shallow-solution behavior through repeated depth-first limited searches.

C - Incorrect: No greedy heuristic ordering is required for IDS.

D - Incorrect: DLS is used, but UCS ordering is not part of standard iterative deepening.

Primary note proof

Study-note section AI16

DLS runs DFS to depth l and reports cutoff separately from failure. It misses goals deeper than l. Iterative deepening runs DLS with limits 0,1,2,...; for finite b it finds the shallowest solution with O(b^d) time and O(bd) frontier space, optimal for equal step costs. Repeated shallow work is small compared with the deepest level.

Study-note section AI13

BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Question sources

Artificial Intelligence review questions · Question 14

Question 189 | NS0015

A zero-sum game evaluator measures a position from the designated MAX player's perspective. What does a positive evaluation value indicate?

A. A draw or neutral position.

B. An estimated advantage for the designated MAX player.

C. An estimated advantage for the opposing MIN player.

D. An invalid game state.

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: An estimated advantage for the designated MAX player.

Under this convention, positive values estimate an advantage for MAX, negative values favor MIN, and zero is neutral. A positive heuristic score does not guarantee an eventual win, because it may be only an estimate at a search cutoff. The reference player must be specified before a sign can be interpreted; player names alone do not determine the convention.

Why each choice is right or wrong

A - Incorrect: A neutral result is represented by zero under the stated sign convention.

B - Correct: A positive value favors the designated MAX player under this convention.

C - Incorrect: An advantage for MIN would have a negative sign under this convention.

D - Incorrect: The sign estimates desirability, not whether the position is legally valid.

Primary note proof

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Question sources

Artificial Intelligence review questions · Question 15

Question 190 | NS0016

Which knowledge representation technique uses interconnected nodes and arcs to show concept relationships?

A. Frames

B. Logic

C. Semantic Networks

D. Probabilistic Models

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Semantic networks.

AI24 defines semantic networks using concept nodes and labeled relations such as is-a. That matches the node/arc representation in the stem. Frames organize slots, logic uses formal sentences and probabilistic models attach uncertainty; those may have related graph forms, but the specific concept-relationship network term is semantic network.

Why each choice is right or wrong

A - Incorrect: Frames represent stereotyped entities through slots and values rather than this named concept-link network.

B - Incorrect: Logic represents claims with formal expressions, not specifically the node/arc scheme described.

C - Correct: Semantic networks connect concept nodes with labeled relationship arcs.

D - Incorrect: Probabilistic models focus on uncertainty rather than the specific semantic-relation representation requested.

Primary note proof

Study-note section AI24

> A semantic network connects concept nodes by labeled relations such as is-a. Frames describe stereotyped entities through slots, values and defaults. Production systems use IF conditions THEN conclusions or actions, matching rules against working memory and selecting among applicable rules. A graph suits relationships such as frequently bought together; frames suit attributes. No representation is always best for recommendation without task requirements.

Study-note section AI24

> Knowledge representation encodes facts, objects, relations, events and rules so a machine can use them. Syntax defines legal expressions; semantics defines their meaning. A knowledge base stores assertions; an inference system derives consequences. Declarative knowledge says what is true; procedural knowledge specifies how to act.

Study-note section AI29

A Bayesian network is a directed acyclic graph with conditional probability tables; its joint distribution factors as the product of each node's probability given its parents. A Markov model assumes the next state depends on the current state rather than the entire past. These represent uncertainty; fuzzy membership represents a different issue.

Question sources

Artificial Intelligence review questions · Question 16

Question 191 | NS0017

Which listed group comprises unstructured input modalities from which deep neural models can learn hierarchical features?

A. Tabular numerical data only

B. Structured relational data

C. Unstructured data like images, sound, and text

D. Small labeled datasets

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: Unstructured data like images, sound, and text

Deep learning uses multiple representation layers and can learn features from images, sound and text. Those are the unstructured modalities identified here. Deep models can also process other data, so this is not a claim of universal superiority over every alternative model. Structured relational data is a different organization, and small labeled datasets describe size and supervision rather than modality.

Why each choice is right or wrong

A - Incorrect: Tabular numerical data only excludes the non-tabular modalities requested and is not an inherent deep-learning restriction.

B - Incorrect: Structured relational data is not the unstructured modality group asked for.

C - Correct: Images, sound and text are explicit modalities for learned multilayer representations.

D - Incorrect: A small labeled dataset describes sample size and feedback, not an input modality group.

Primary note proof

Study-note section AI34

Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs

share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Study-note section AI35

Overfitting means fitting training-specific details so performance drops on unseen data; underfitting means the model is too limited or poorly fitted even on training data. A learning curve plots performance against training size or training progress. Complexity control, regularization, pruning and more representative data can help.

Question sources

Artificial Intelligence review questions · Question 17

Question 192 | NS0018

Which agent type improves its performance over time through experience and feedback?

A. Simple Reflex Agent

B. Model-Based Reflex Agent

C. Goal-Based Agent

D. Learning Agent

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer D: A learning agent.

AI08 defines the learning element as changing the agent to improve future performance from experience and feedback. This identifies the learning-agent category. Reflex, model-based and goal-based describe decision architectures that may be enhanced by learning, but do not by themselves require improvement over time; D explicitly names that capability.

Why each choice is right or wrong

A - Incorrect: Simple reflex rules alone do not require adaptation from experience.

B - Incorrect: Maintaining a model does not by itself require learning or improving it over time.

C - Incorrect: Planning toward a goal does not itself establish experience-based policy improvement.

D - Correct: Learning agents explicitly use experience and evaluative feedback to improve future behavior.

Primary note proof

Study-note section AI08

> The performance element selects actions. The learning element changes the agent to improve future performance. The critic evaluates behavior against a performance standard and supplies feedback. The problem generator suggests exploratory actions likely to yield useful experience.

Study-note section AI08

> Learning is improvement on task T, measured by P, through experience E. For spam filtering, T is classifying email, E is past labeled messages, and P might be recall at a required precision. Exploration trades immediate reward for information; exploitation uses the best action currently known. Learning can improve reflex, goal-based or utility-based agents.

Question sources

Artificial Intelligence review questions · Question 18

Question 193 | NS0019

What is abductive reasoning best described as?

A. Drawing certain conclusions from known facts

B. Making educated guesses from patterns

C. Finding the best explanation for a set of observations

D. Using probability to decide outcomes

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer C: Finding a plausible best explanation for observations.

AI28 illustrates abduction with wet ground suggesting rain while allowing sprinklers as another cause. It proposes an explanatory hypothesis rather than guaranteeing truth or merely generalizing a pattern. C gives that explanatory role. Probability may rank hypotheses, but numerical probability alone is not the definition of abductive reasoning.

Why each choice is right or wrong

A - Incorrect: Certain truth-preserving conclusions describe valid deduction rather than explanatory abduction.

B - Incorrect: Pattern-based generalization is closer to induction and omits the explanatory-hypothesis requirement.

C - Correct: Abduction seeks a plausible explanation for observed evidence without guaranteeing that explanation.

D - Incorrect: Probability can support a diagnosis but does not itself define the explanatory reasoning form.

Primary note proof

Study-note section AI28

> Deduction applies truth-preserving rules: all people are mortal; Ada is a person; therefore Ada is mortal. Induction generalizes observations to a broader rule and can be overturned by new examples. Abduction proposes a plausible explanation: wet ground suggests rain, although sprinklers could explain it. Analogy transfers a pattern between similar situations.

Study-note section AI29

> Probability describes uncertainty about events. $P(\neg A)=1-P(A)$; $P(A|B)=P(A\cap B)/P(B)$, for $P(B)>0$. Bayes gives $P(H|E)=P(E|H)P(H)/P(E)$: posterior equals likelihood times prior divided by evidence probability. For mutually exclusive exhaustive hypotheses, $P(E)=\Sigma_i P(E|Hi)P(Hi)$.

Question sources

Artificial Intelligence review questions · Question 19

Question 194 | NS0020

How is ordinary untimed chess classified with respect to transition outcomes, change during deliberation and state/action values? Assume both players' moves are specified when evaluating a transition.

A. Stochastic, Dynamic, Continuous

B. Deterministic, Static, Discrete

C. Stochastic, Static, Discrete

D. Deterministic, Dynamic, Continuous

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Deterministic, Static, Discrete

Chess has discrete positions and legal moves. Given the selected move, the resulting position is determined by the rules rather than random outcome sampling. In untimed play, the environment does not change while the player deliberates, so the relevant classification is static. Opponent choice is strategic uncertainty rather than dice-like stochasticity. Specifying untimed play excludes the semidynamic effect of a score-affecting clock.

Why each choice is right or wrong

A - Incorrect: Chess moves are discrete, and opponent strategy is not dice-like stochasticity.

B - Correct: Untimed chess is deterministic given chosen moves, static while deliberating and discrete.

C - Incorrect: Ordinary chess transition outcomes are deterministic given chosen moves rather than stochastic.

D - Incorrect: Chess is discrete rather than continuous, and a clock gives semidynamic rather than this classification.

Primary note proof

Study-note section AI09

> Fully observable means sensors reveal relevant state; partially observable means relevant information is hidden or noisy. Deterministic means state and action determine the next state; stochastic means outcomes involve randomness. Strategic uncertainty about an opponent is distinct from dice-like randomness.

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Question sources

Artificial Intelligence review questions · Question 20

Question 195 | NS0021

What does the 'deep' in Deep Learning refer to?

A. The depth of the training dataset

B. The depth of layers in a neural network

C. The complexity of the problem being solved

D. The deep analysis of heuristic functions

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Multiple neural representation layers.

AI34 defines deep learning through multiple representation layers in a neural model. Dataset size and problem difficulty can affect training, but neither is what the word deep denotes. Heuristic analysis belongs to search methods rather than the architecture depth of a neural network.

Why each choice is right or wrong

A - Incorrect: Dataset size or depth does not define the deep part of deep learning.

B - Correct: Deep refers to multiple layers of learned representations in the neural architecture.

C - Incorrect: Problem difficulty does not determine whether a model architecture is deep.

D - Incorrect: Heuristic analysis is not the neural-layer definition of deep learning.

Primary note proof

Study-note section AI34

> Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Question sources

Artificial Intelligence review questions · Question 21

Question 196 | NS0022

Which search strategy is called 'blind search' because it uses no extra problem knowledge?

A. Informed Search

B. Heuristic Search

C. Uninformed Search

D. Best-First Search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Uninformed search.

AI17 says uninformed methods use the problem definition and known costs without a problem-specific remaining-cost estimate. This is the usual blind-search meaning. Informed or heuristic search explicitly adds such estimates, and best-first is a broad ordering family that may use heuristic information rather than naming the blind category.

Why each choice is right or wrong

A - Incorrect: Informed search adds problem-specific guidance beyond the basic problem definition.

B - Incorrect: Heuristic search uses the extra estimate excluded by the stem.

C - Correct: Uninformed search is the category using only the problem definition and known costs.

D - Incorrect: Best-first specifies evaluation-based ordering and is not generally synonymous with blind search.

Primary note proof

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI15

> UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Question sources

Artificial Intelligence review questions · Question 22

Question 197 | NS0023

In Reinforcement Learning, what guides an agent to improve its behavior?

A. Labeled training examples

B. Hard-coded rules

C. Rewards and penalties from environment interactions

D. Pre-trained heuristic functions

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: Reward feedback from interactions.

AI33 describes an agent acting, receiving reward and a new observation, and improving its policy to maximize expected cumulative reward. Rewards may be positive or negative and delayed; they need not name the correct action. Labeled examples, fixed rules and predefined heuristics do not provide this distinctive reinforcement feedback mechanism.

Why each choice is right or wrong

A - Incorrect: Correct target labels characterize supervised learning rather than reinforcement feedback.

B - Incorrect: Fixed rules alone do not provide the reward-based adaptation defining RL.

C - Correct: Rewards and penalties evaluate outcomes and guide improvement of expected cumulative return.

D - Incorrect: A pretrained heuristic may help implementation but is not the defining RL feedback signal.

Primary note proof

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Question sources

Artificial Intelligence review questions · Question 23

Question 198 | NS0024

For greedy best-first search on an implicit finite tree with branching factor b and maximum depth m, what is the standard worst-case node-count time bound?

A. O(m)

B. O(b*m)

C. O(b^m)

D. O(log m)

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: O(b^m)

Greedy best-first ordering can be misled by a heuristic and may need to explore an exponentially large portion of the finite search tree. With branching factor b and depth m, the standard worst-case generated-node bound is O(b^m). This is a node-count convention rather than a claim about every priority-queue implementation or all best-first algorithms. The explicit algorithm and tree model determine the bound.

Why each choice is right or wrong

A - Incorrect: A misleading heuristic can force exponentially many nodes, so a bound merely linear in maximum depth m is insufficient.

B - Incorrect: O(bm) is a familiar DFS storage form, not this generic search-time guarantee.

C - Correct: The specified implicit finite-tree worst case can require O(b^m) nodes.

D - Incorrect: A logarithmic-in-depth bound is insufficient when the search can explore an exponential number of tree nodes.

Primary note proof

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Question sources

Artificial Intelligence review questions · Question 24

Question 199 | NS0025

Which knowledge representation uses templates with slots to describe stereotyped situations?

A. Semantic Networks

B. Frames

C. Logic

D. Bayesian Networks

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: Frames.

AI24 defines frames as stereotyped entity descriptions with slots, values and defaults. This is the template-and-slot representation in the question. Semantic networks focus on relationships among concept nodes, formal logic expresses sentences and Bayesian networks represent probabilistic dependencies; none is the specific slot-template term requested.

Why each choice is right or wrong

A - Incorrect: Semantic networks emphasize concept nodes and labeled relations rather than the slot-template representation.

B - Correct: Frames organize stereotyped descriptions into slots with values and defaults.

C - Incorrect: Logic uses formal sentences rather than the specifically described frame slots.

D - Incorrect: Bayesian networks represent conditional probabilities, not the named stereotyped slot template.

Primary note proof

Study-note section AI24

> A semantic network connects concept nodes by labeled relations such as is-a. Frames describe stereotyped entities through slots, values and defaults. Production systems use IF conditions THEN conclusions or actions, matching rules against working memory and selecting among applicable rules. A graph suits relationships such as frequently bought together; frames suit attributes. No representation is always best for recommendation without task requirements.

Study-note section AI29

> A Bayesian network is a directed acyclic graph with conditional probability tables; its joint distribution factors as the product of each node's probability given its parents. A Markov model assumes the next state depends on the current state rather than the entire past. These represent uncertainty; fuzzy membership represents a different issue.

Question sources

Artificial Intelligence review questions · Question 25

Question 200 | NS0026

What is Natural Language Processing (NLP) primarily focused on?

A. Visual image interpretation

B. Physical robot movement

C. Enabling machines to understand, interpret, and generate human language

D. Optimizing database queries

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer C: Understanding, interpreting and generating human language.

AI37 directly defines NLP as interpreting and generating human language and lists dialogue, translation and summarization. Visual interpretation belongs to computer vision, physical movement to robotics and query optimization to database processing. NLP can interact with those systems, but its primary language-processing domain remains distinct.

Why each choice is right or wrong

A - Incorrect: Interpreting visual images is primarily computer vision rather than language processing.

B - Incorrect: Physical robot movement concerns control and robotics, not the definition of NLP.

C - Correct: Human-language interpretation and generation are the primary NLP tasks named in AI37.

D - Incorrect: Database-query optimization is not the human-language-processing focus described.

Primary note proof

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI39

> Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and

efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Question sources

Artificial Intelligence review questions · Question 26

Question 201 | NS0027

Which listed term is a reasoning method rather than one of the common learning regimes classified by availability of target labels?

A. Supervised Learning

B. Unsupervised Learning

C. Deductive reasoning

D. Semi-supervised Learning

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer C: Deductive reasoning

Supervised, unsupervised and semi-supervised learning differ in the target labels available during training. Deductive reasoning instead derives consequences from premises using truth-preserving rules. Deduction can be used inside a learning system, so the distinction is not that the two can never coexist. The question separates a reasoning method from the conventional label-availability taxonomy.

Why each choice is right or wrong

A - Incorrect: Supervised learning uses labeled input-target pairs, so it is a label-availability learning regime.

B - Incorrect: Unsupervised learning has no supplied targets and discovers structure, so it is a label-availability regime.

C - Correct: Deduction derives truth-preserving consequences from premises rather than naming a label-availability regime.

D - Incorrect: Semi-supervised learning combines labeled and unlabeled data, so it is a label-availability regime.

Primary note proof

Study-note section AI31

Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI32

Semi-supervised learning combines some labels with additional unlabeled data. Active learning chooses informative cases for an oracle to label; it still uses labels. “Exploratory learning” is not an exclusive standard synonym: it can mean unsupervised exploration or active data acquisition, so define the feedback. A task named summarization or time-series analysis is not automatically supervised; its training targets determine that.

Study-note section AI28

Deduction applies truth-preserving rules: all people are mortal; Ada is a person; therefore Ada is mortal. Induction generalizes observations to a broader rule and can be overturned by new examples. Abduction proposes a plausible explanation: wet ground suggests rain, although sprinklers could explain it. Analogy transfers a pattern between similar situations.

Question sources

Artificial Intelligence review questions · Question 27

Question 202 | NS0028

In adversarial search, what makes it different from traditional search?

A. It only uses one agent

B. Multiple agents with conflicting goals compete in the same search space

C. It guarantees the optimal solution always

D. It does not use evaluation functions

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: Competing agents with conflicting objectives.

AI22 defines adversarial search by agents with competing objectives and gives minimax for opposition. That distinguishes it from a single-agent path search. It can use evaluation functions at cutoffs, and its best-outcome guarantee is conditional on the modeled game and opponent, not an unconditional guarantee that every implementation finds a globally optimal solution.

Why each choice is right or wrong

A - Incorrect: A single decision-making agent does not capture the competing-objective feature of adversarial search.

B - Correct: Competing agents and conflicting objectives are its defining distinction.

C - Incorrect: Guarantees depend on the algorithm, model and search depth, not merely the adversarial label.

D - Incorrect: Depth-limited adversarial search commonly uses heuristic evaluation functions.

Primary note proof

Study-note section AI22

> Adversarial search models agents with competing objectives. A game specifies states, player to move, legal actions, successor function, terminal test and utility. Perfect information means previous moves and current state are visible; it does not reveal future choices or imply absence of chance. Zero-sum utility makes one player's gain the other's loss.

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Question sources

Artificial Intelligence review questions · Question 28

Question 203 | NS0029

A Depth-Limited Search (DLS) is NOT complete in itself because:

A. It uses too much memory

B. Solutions beyond the preset depth limit will be missed

C. It explores nodes in random order

D. It requires a heuristic function

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: A solution beyond the fixed depth limit is missed.

AI16 defines DLS as DFS stopped at depth l and explicitly says it misses goals deeper than that. A fixed limit therefore does not establish completeness for arbitrary problem depths. This is independent of randomized ordering, heuristics or memory use; increasing limits through iterative deepening addresses the depth restriction under its stated assumptions.

Why each choice is right or wrong

A - Incorrect: Depth-limited DFS is memory-efficient; excessive memory is not the reason goals are missed.

B - Correct: A preset cutoff prevents exploring a solution whose depth exceeds that limit.

C - Incorrect: DLS need not use random order; the depth cutoff alone causes the limitation.

D - Incorrect: No heuristic is required for depth-limited DFS.

Primary note proof

Study-note section AI16

> DLS runs DFS to depth l and reports cutoff separately from failure. It misses goals deeper than l. Iterative deepening runs DLS with limits 0,1,2,...; for finite b it finds the shallowest solution with O(b^d) time and O(bd) frontier space, optimal for equal step costs. Repeated shallow work is small compared with the deepest level.

Study-note section AI14

> DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Question sources

Artificial Intelligence review questions · Question 29

Question 204 | NS0030

Which probabilistic model represents knowledge in terms of conditional probabilities among variables?

A. Semantic Networks

B. Frames

C. Bayesian Networks

D. Ontologies

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Bayesian networks.

AI29 defines a Bayesian network as a DAG with conditional probability tables and joint factorization by parents. That is the probabilistic knowledge representation in the stem. Semantic networks and frames encode relations or slots without inherently supplying conditional distributions, and ontologies specify conceptual structure rather than this particular probabilistic factorization.

Why each choice is right or wrong

A - Incorrect: Semantic networks encode labeled concept relations rather than inherently conditional probability distributions.

B - Incorrect: Frames encode slots, values and defaults rather than the named probabilistic dependency structure.

C - Correct: Bayesian networks explicitly store conditional probability relationships among variables.

D - Incorrect: An ontology's conceptual organization is not itself the conditional-probability model specified.

Primary note proof

Study-note section AI29

> A Bayesian network is a directed acyclic graph with conditional probability tables; its joint distribution factors as the product of each node's probability given its parents. A Markov model assumes the next state depends on the current state rather than the entire past. These represent uncertainty; fuzzy membership represents a different issue.

Study-note section AI24

> A semantic network connects concept nodes by labeled relations such as is-a. Frames describe stereotyped entities through slots, values and defaults. Production systems use IF conditions THEN conclusions or actions, matching rules against working memory and selecting among applicable rules. A graph suits relationships such as frequently bought together; frames suit attributes. No representation is always best for recommendation without task requirements.

Study-note section AI-S06

An ontology describes entities or individuals, classes and properties and may state logical relationships and restrictions. It organizes explicit conceptual knowledge. Conditional probability tables and a probabilistic factorization are additional modeling commitments, not inherent in the word ontology.

Question sources

Artificial Intelligence review questions · Question 30

Question 205 | NS0031

What is an 'expanded node' in a search algorithm?

A. A node that has been visited but not explored further

B. A node selected for further exploration whose neighboring nodes have been generated

C. A node removed from the open list

D. A node that has reached the goal state

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: A node whose successors have been generated.

AI12 defines expansion as generating successors. B states that action, unlike mere visitation, removal from OPEN or reaching a goal. A node can be removed and goal-tested without expanding it, and discovered frontier nodes await expansion. The criterion is what was generated, not whether the node was seen or selected.

Why each choice is right or wrong

A - Incorrect: Visiting without generating successors does not meet the definition of expansion.

B - Correct: Generating the selected node’s neighboring successor nodes is expansion.

C - Incorrect: Removal alone may precede a goal test without generating any successors.

D - Incorrect: Goal status is not the definition of expansion; a goal may be accepted without expanding it.

Primary note proof

Study-note section AI12

> A search node records a state plus parent, action, depth and path cost. Expanding a node generates its successors. OPEN or frontier stores discovered candidates awaiting expansion; CLOSED records expanded states. Reaching the same state through different paths does not imply the same node or cost.

Study-note section AI13

> BFS removes the oldest frontier entry using a FIFO queue. Start with the root, remove the head, test it, append unseen successors at the tail, and repeat. It explores depths 0,1,2,... and is complete for finite b and a finite-depth goal. It minimizes number of edges, so minimum cost requires equal positive step costs.

Question sources

Artificial Intelligence review questions · Question 31

Question 206 | NS0032

For which type of problem would Supervised Learning with CNNs be most appropriate?

A. Customer segmentation from unlabeled data

B. Classifying a large labeled image dataset

C. Learning chess actions from rewards obtained through trial-and-error play.

D. Finding hidden patterns in transaction data

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Classifying a large labeled image dataset

The labeled images provide input-target pairs for supervised classification, and CNN filters exploit spatial image structure. Unlabeled customer segmentation and transaction-pattern discovery seek structure without class targets. Learning chess actions from rewards is a reinforcement objective.

Why each choice is right or wrong

A - Incorrect: Unlabeled segmentation seeks groups rather than supervised image classes.

B - Correct: Labeled images supply class targets, and CNN filters learn spatial image features.

C - Incorrect: Trial-and-error chess with rewards defines a reinforcement objective rather than supplied image-class targets.

D - Incorrect: Hidden transaction-pattern discovery seeks unlabeled structure rather than labeled image classification.

Primary note proof

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI34

> Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Study-note section AI32

Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Question sources

Artificial Intelligence review questions · Question 32

Question 207 | NS0033

What does 'perfect information' mean in the context of game theory?

A. The agent knows the future

B. Both players have complete and accurate knowledge of the entire game state

C. The game has no randomness

D. Each player can see only their own pieces

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: The current game state and previous moves are visible to the players.

AI22 defines perfect information as visibility of previous moves and the current state. This makes B the correct state-knowledge description. It does not reveal future choices and does not by itself remove chance from a game. Seeing only one's own pieces instead describes hidden relevant state, inconsistent with perfect information.

Why each choice is right or wrong

A - Incorrect: Future choices are not known merely because the present game state is visible.

B - Correct: Complete visibility of current state and relevant move history is the intended information condition.

C - Incorrect: A game can include chance while still revealing the state and prior moves.

D - Incorrect: Seeing only one's own pieces leaves relevant state hidden and is not perfect information.

Primary note proof

Study-note section AI22

> Adversarial search models agents with competing objectives. A game specifies states, player to move, legal actions, successor function, terminal test and utility. Perfect information means previous moves and current state are visible; it does not reveal future choices or imply absence of chance. Zero-sum utility makes one player's gain the other's loss.

Study-note section AI04

> Omniscience would mean knowing actual action outcomes in advance. Rationality does not require omniscience or guaranteed success. Even complete outcome knowledge cannot create a successful action when every available action fails. Information gathering and exploration may therefore be rational actions.

Question sources

Artificial Intelligence review questions · Question 33

Question 208 | NS0034

Semi-supervised Learning is typically used when:

A. All data is labeled

B. No data is labeled

C. Obtaining a fully labeled dataset is time-consuming or costly

D. The problem requires reinforcement feedback

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer C: When labels are costly and some labeled data can be supplemented by unlabeled data.

AI32 defines semi-supervised learning as combining some labels with additional unlabeled data. Costly full labeling provides the motivation described in C, provided at least some labeled targets exist. Fully labeled data is ordinary supervised learning; wholly unlabeled structure fits unsupervised learning, and environmental rewards specify reinforcement feedback rather than this mixed dataset regime.

Why each choice is right or wrong

A - Incorrect: A fully labeled dataset does not give the defining labeled-plus-unlabeled mixture.

B - Incorrect: With no labels at all, the described regime is unsupervised rather than semi-supervised.

C - Correct: Expensive labeling motivates using a limited labeled subset together with plentiful unlabeled examples.

D - Incorrect: Reward feedback defines reinforcement learning rather than a mixture of labeled and unlabeled examples.

Primary note proof

Study-note section AI32

> Semi-supervised learning combines some labels with additional unlabeled data. Active learning chooses informative cases for an oracle to label; it still uses labels. “Exploratory learning” is not an exclusive standard synonym: it can mean unsupervised exploration or active data acquisition, so define the feedback. A task named summarization or time-series analysis is not automatically supervised; its training targets determine that.

Study-note section AI31

> Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Question sources

Artificial Intelligence review questions · Question 34

Question 209 | NS0035

In the context of AI agents, what are 'effectors'?

A. Sensors that gather environmental data

B. Components that allow an agent to take actions in its environment

C. Internal memory modules

D. Heuristic evaluation functions

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Components through which an agent acts on its environment.

AI03 uses actuators or effectors for acting components and sensors for perception. Therefore B gives the correct role. Memory stores internal information and heuristic functions estimate promise; neither is the acting interface. An effector may be physical hardware or a software action mechanism depending on the agent’s architecture.

Why each choice is right or wrong

A - Incorrect: Sensors gather percepts; effectors implement actions rather than observations.

B - Correct: Effectors are the components through which selected actions affect the environment.

C - Incorrect: Internal memory maintains information but is not the general acting interface.

D - Incorrect: Heuristic functions evaluate promise and do not themselves execute environmental actions.

Primary note proof

Study-note section AI03

> An agent perceives an environment through sensors and acts through actuators or effectors. Cameras sense; motors move. Software sensors may be incoming messages and software actions may be API requests. Humans are agents but are not artificial agents.

Study-note section AI06

A model-based reflex agent maintains internal state using the previous state, previous action, current percept, and models of world change and action effects. It then applies rules to that state. Both kinds can use condition-action rules; memory and a world model distinguish them.

Study-note section AI17

A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Question sources

Artificial Intelligence review questions · Question 35

Question 210 | NS0036

Which of the following best describes a Utility-Based Agent?

A. It only reacts to current perceptions using condition-action rules

B. It selects actions that maximize expected utility, handling conflicting goals through trade-offs

C. It plans sequences of actions toward a single fixed goal

D. It learns entirely from environmental feedback

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Choose actions with maximum expected utility, balancing outcome trade-offs.

AI07 defines utility-based agents by comparing desirability and trade-offs and maximizing expected utility over uncertain outcomes. This is broader than merely satisfying a fixed goal. Current-percept rule matching describes simple reflex behavior, and experience-based improvement describes learning, which may coexist with utility choice but does not define it.

Why each choice is right or wrong

A - Incorrect: Current-percept condition-action matching describes a simple reflex architecture.

B - Correct: Expected-utility maximization explicitly compares trade-offs among uncertain outcomes.

C - Incorrect: Goal-reaching planning alone does not rank degrees of desirability among outcomes.

D - Incorrect: Learning describes adaptation from feedback, not the utility-based decision criterion itself.

Primary note proof

Study-note section AI07

A goal-based agent considers future action sequences that reach a desired state. A utility-based agent compares degrees of desirability and trade-offs, choosing maximum expected utility: $EU(a)=\Sigma s\ P(s|a,evidence)U(s)$. Reaching the destination is a goal; balancing arrival time, safety and cost requires preferences among outcomes.

Study-note section AI06

A simple reflex agent matches condition-action rules against the current percept only: if dirty, clean. It has no persistent world-state memory. This is adequate when the current observation contains everything needed for its rule, but partial observability can make the same percept require different actions.

Study-note section AI08

The performance element selects actions. The learning element changes the agent to improve future performance. The critic evaluates behavior against a performance standard and supplies feedback. The problem generator suggests exploratory actions likely to yield useful experience.

Question sources

Artificial Intelligence review questions · Question 36

Question 211 | NS0037

What is Computer Vision in AI primarily concerned with?

A. Processing and generating spoken language

B. Enabling machines to understand and interpret visual information

C. Designing physical robotic systems

D. Managing database indexing

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer B: Interpreting visual information from images or video.

AI39 defines computer vision as interpreting images or video and includes classification, detection, segmentation and recognition. Language generation, robot mechanics and database indexing are separate task areas, although vision may be one component of a robot. B states the visual information-processing purpose directly.

Why each choice is right or wrong

A - Incorrect: Speech and language generation belong to language/speech processing rather than visual interpretation.

B - Correct: Computer vision extracts and interprets information from visual inputs such as images and video.

C - Incorrect: Robotic system design may use vision but is broader than the visual-processing field.

D - Incorrect: Database indexing is not the interpretation of visual scenes or objects.

Primary note proof

Study-note section AI39

> Computer vision interprets images or video: classification names image content, detection locates objects, segmentation labels pixels, and recognition identifies objects or people. Occlusion, lighting change, noise, unfamiliar conditions and incomplete context make open-world perception difficult even when a face detector works well.

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI39

> Robotics combines mechanical, electrical, control and computing systems to sense, plan and act physically. Components include sensors, controller, power supply, actuators and an end effector. Locomotion may be legged, wheeled or tracked, with trade-offs in terrain, stability and efficiency. Software can act autonomously without being a robot. Humanoid shape is not evidence of general intelligence.

Question sources

Artificial Intelligence review questions · Question 37

Question 212 | NS0038

In a zero-sum Tic-Tac-Toe terminal utility convention, MAX wins score +1 and MIN wins score −1. Which listed value represents a draw giving neither player an advantage?

A. +1

B. -1

C. +Infinity

D. 0

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer D: 0

The convention assigns equal and opposite utilities to the two decisive outcomes. A draw gives neither side an advantage, so its neutral utility is zero under that symmetric scale. Positive one favors MAX and negative one favors MIN; positive infinity would strongly favor MAX rather than indicate neutrality. This is an explicit utility convention, not an intrinsic numerical property of the game outcome.

Why each choice is right or wrong

A - Incorrect: Under the common stated scale, +1 is a win rather than a draw.

B - Incorrect: Under the common stated scale, −1 is a loss rather than a draw.

C - Incorrect: Positive infinity is not the draw value in the note's common scale.

D - Correct: Zero is the neutral utility between +1 and −1 under this convention.

Primary note proof

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Question sources

Artificial Intelligence review questions · Question 38

Question 213 | NS0039

Treat a whole independently reset chess match as one episode with no effects on later matches; treat the other options as ongoing tasks at the action level described. Which task is episodic under those units?

A. A self-driving car navigating city traffic

B. A chess game (each individual match)

C. A robot cleaning a multi-room house continuously

D. A stock trading algorithm operating over months

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: A chess game (each individual match)

An episodic task has independent decision units. Under the stated convention, each whole chess match is reset and cannot affect later matches, so matches can be treated as episodes. Individual moves within a match remain sequential. Driving, continuous cleaning and continuing stock trading all have actions that affect later situations within the ongoing task. The classification therefore follows the explicit unit and independence assumptions.

Why each choice is right or wrong

A - Incorrect: Driving decisions affect later positions and traffic interactions, making the task sequential.

B - Correct: Whole independently reset matches are independent episodes under the stated convention.

C - Incorrect: Cleaning actions change later room states and decisions in the continuing task.

D - Incorrect: Trading decisions affect later holdings and opportunities, so the continuing task is sequential.

Primary note proof

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Question sources

Artificial Intelligence review questions · Question 39

Question 214 | NS0040

The heuristic function h(n) in A* represents:

A. The exact cost from start to node n

B. The total path cost already traveled

C. The estimated cost from node n to the goal state

D. The branching factor at node n

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: Estimated remaining cost from n to a goal.

AI17 defines h as an estimate of remaining cost or promise, and AI18 adds it to accumulated g in A*. Thus C is the heuristic component. Exact start-to-node cost belongs to g, and branching factor b is a search-size parameter rather than a cost-to-go estimate.

Why each choice is right or wrong

A - Incorrect: Exact cost from the start belongs to g(n), not the heuristic h(n).

B - Incorrect: Already-traveled path cost is g(n), not estimated future cost.

C - Correct: h estimates the remaining cost from the current node to a goal.

D - Incorrect: Branching factor counts successor alternatives and is not the heuristic cost estimate.

Primary note proof

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI18

> A* orders by f(n)=g(n)+h(n). Admissibility means $0 \le h(n) \le h^*(n)$, the true cheapest remaining cost. Consistency means $h(n) \le c(n,n') + h(n')$ on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI12

> Completeness means finding a solution if one exists under stated assumptions; optimality means returning the best solution. Time counts work, often generated nodes; space counts simultaneous storage. Let b be branching factor, d shallowest goal depth, m maximum search depth, and l a depth limit. Graph-search bounds in vertices V and edges E differ from implicit-tree bounds.

Question sources

Artificial Intelligence review questions · Question 40

Question 215 | NS0041

When a doctor proposes a disease as a plausible cause explaining observed symptoms, which term names this explanatory direction of reasoning, rather than a method for numerically ranking hypotheses?

A. Deductive Reasoning

B. Inductive Reasoning

C. Abductive Reasoning

D. Probabilistic Reasoning

Hint

Translate the statement carefully, then check the scope of its quantifiers, connectives or inference rule. Distinguish guaranteed consequences from plausible generalizations or explanations.

Answer C: Abductive Reasoning

Abduction proposes an explanation for observed evidence, such as a disease that could account for symptoms. Other causes may also fit, so the conclusion is not logically guaranteed. Probability can help rank the possible explanations, but that numerical method is distinct from the explanatory direction. Deduction preserves truth from premises, and induction generalizes patterns across cases.

Why each choice is right or wrong

A - Incorrect: Deduction follows truth-preserving implications from premises, not a merely plausible cause inferred from its effects.

B - Incorrect: Induction generalizes across observations rather than specifically proposing a cause that explains an observation.

C - Correct: Abduction proposes a plausible explanation for observed evidence.

D - Incorrect: Probability can rank disease hypotheses, but does not itself name the observation-to-explanation direction.

Primary note proof

Study-note section AI28

> Deduction applies truth-preserving rules: all people are mortal; Ada is a person; therefore Ada is mortal. Induction generalizes observations to a broader rule and can be overturned by new examples. Abduction proposes a plausible explanation: wet ground suggests rain, although sprinklers could explain it. Analogy transfers a pattern between similar situations.

Study-note section AI29

> Probability describes uncertainty about events. $P(\neg A)=1-P(A)$; $P(A|B)=P(A\cap B)/P(B)$, for $P(B)>0$. Bayes gives $P(H|E)=P(E|H)P(H)/P(E)$: posterior equals likelihood times prior divided by evidence probability. For mutually exclusive exhaustive hypotheses, $P(E)=\Sigma_i P(E|Hi)P(Hi)$.

Question sources

Artificial Intelligence review questions · Question 41

Question 216 | NS0042

What major ethical concern does AI raise related to decision-making systems?

A. AI systems use too little data

B. Bias in algorithms leading to unfair or discriminatory outcomes

C. AI always makes perfectly rational decisions

D. AI cannot process unstructured data

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Biased decisions causing unfair outcomes.

AI40 says bias in data or evaluation can create unfair outcomes and calls for checking errors across groups, robustness and oversight. B states the ethical concern directly. Data scarcity may be a technical issue, but little data is not the stated general ethical principle; perfect rationality and inability to process unstructured inputs are false generalizations.

Why each choice is right or wrong

A - Incorrect: Using too little data is not the specific fairness concern described here.

B - Correct: Biased data or evaluation can produce unfair or discriminatory decisions.

C - Incorrect: AI is not guaranteed perfect rational decisions or error-free outcomes.

D - Incorrect: The note explicitly includes models that process images, sound and text.

Primary note proof

Study-note section AI40

> Rules offer inspectable domain reasoning but can be brittle and costly to maintain. Learned models handle data-rich patterns but depend on representative data and validation. Search suits explicit action alternatives; probabilistic models suit uncertain evidence. Bias in data or evaluation can create unfair outcomes. Evaluate errors across relevant groups, robustness, privacy and human oversight rather than assuming automation is always beneficial.

Study-note section AI34

> Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Study-note section AI04

A performance measure defines success externally: safety, useful deliveries, elapsed time, or cost. Rationality depends on that measure, prior knowledge, percept history, and available actions. A rational agent selects the action with the best expected performance given its information.

Question sources

Artificial Intelligence review questions · Question 42

Question 217 | NS0043

In a continuous environment like self-driving cars, what challenge does 'continuous' specifically pose?

A. A finite set of states makes decisions trivial

B. Infinite or large state/action spaces require more complex computations

C. The environment never changes

D. All outcomes are deterministic

Hint

Read the relevant agent component, action-selection criterion or environment dimension. Use only properties established by the description.

Answer B: Real-valued state/action variables create large or infinite spaces to represent and search.

AI09 defines continuous quantities as varying along real-valued ranges, unlike distinct discrete alternatives. Such ranges can create unboundedly many state/action possibilities, motivating approximation or richer computation rather than exhaustive finite enumeration. Continuity is separate from whether a world changes over time or whether outcomes are stochastic.

Why each choice is right or wrong

A - Incorrect: Continuous ranges do not reduce the task to a trivial finite set of alternatives.

B - Correct: Real-valued ranges can produce very large or infinite possibilities needing suitable representation and computation.

C - Incorrect: No world change defines static rather than continuous.

D - Incorrect: Determinism is a separate transition-property dimension and does not follow from continuity.

Primary note proof

Study-note section AI09

> Episodic decisions are independent; sequential decisions affect later situations. Static worlds do not change while deliberating; dynamic ones do. A clock affecting score makes an otherwise static game semidynamic. Discrete quantities have distinct alternatives; continuous quantities vary along real-valued ranges. Known versus unknown concerns knowledge of the transition model, not observability.

Question sources

Artificial Intelligence review questions · Question 43

Question 218 | NS0044

The successor function in game search is responsible for:

A. Evaluating terminal game states

B. Generating new states by applying legal moves to the current game state

C. Pruning unpromising branches

D. Calculating the heuristic value of each state

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: Generate successor states by applying legal moves.

AI22 lists legal actions and a successor function as game components; AI11 identifies successors as states reached by actions. In game search, applying each legal move yields its successor position. Terminal utilities and heuristic scores evaluate positions, while alpha-beta pruning avoids exploring branches; neither is the transition-generation function.

Why each choice is right or wrong

A - Incorrect: Terminal utility evaluation scores a completed outcome rather than generating the next position.

B - Correct: The successor function maps legal moves at the current position to resulting game states.

C - Incorrect: Pruning skips exploration under a bound; it is not state-transition generation.

D - Incorrect: A heuristic evaluation scores a state's promise rather than producing legal successor states.

Primary note proof

Study-note section AI22

> Adversarial search models agents with competing objectives. A game specifies states, player to move, legal actions, successor function, terminal test and utility. Perfect information means previous moves and current state are visible; it does not reveal future choices or imply absence of chance. Zero-sum utility makes one player's gain the other's loss.

Study-note section AI11

> First formulate the goal; then formulate the problem by specifying initial state, legal actions, transition model, goal test, and action costs. Search selects a sequence; execution applies it. A state abstracts away irrelevant world details.

Study-note section AI23

> At a depth cutoff use an evaluation function estimating position desirability. Positive favors the designated MAX player, negative MIN; the scale is a convention, with +1/0/−1 often win/draw/loss. Material counts weighted pieces; mobility counts legal options; territory scores

controlled regions. These are different features, often combined. Heuristics improve limited computation, not guarantee victory.

Question sources

Artificial Intelligence review questions · Question 44

Question 219 | NS0045

For implicit-tree search with branching factor b and shallowest goal depth d, which is a key memory advantage of iterative deepening over BFS when each depth-limited pass stores pending siblings?

A. It finds deeper solutions faster

B. It requires storing all nodes at each level in memory

C. It uses O(bd) frontier space rather than storing an entire BFS level.

D. It does not guarantee completeness

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer C: It uses O(bd) frontier space rather than storing an entire BFS level.

Iterative deepening repeats depth-limited DFS while increasing the limit. A pass retains a depth-first path and pending siblings, giving O(bd) frontier space rather than BFS's potentially exponential level frontier. It remains complete for finite branching and a finite-depth solution, although it repeats some shallow work. The option states the correct memory advantage without claiming that every implementation stores only the current path.

Why each choice is right or wrong

A - Incorrect: IDS repeats shallow work and does not universally find deeper solutions faster than BFS.

B - Incorrect: Storing all nodes at a level is BFS-style memory behavior, not the IDS advantage.

C - Correct: The stated implementation keeps only a depth-first frontier, requiring O(bd) rather than the BFS level frontier.

D - Incorrect: IDS is complete under finite branching and a finite-depth solution, so loss of completeness is not its benefit.

Primary note proof

Study-note section AI16

DLS runs DFS to depth l and reports cutoff separately from failure. It misses goals deeper than l. Iterative deepening runs DLS with limits 0,1,2,...; for finite b it finds the shallowest solution with O(b^d) time and O(bd) frontier space, optimal for equal step costs. Repeated shallow work is small compared with the deepest level.

Study-note section AI13

The tree contains 1+b+...+b^d nodes through depth d, giving O(b^d) under generation-time goal testing. Expanding depth-d nodes before testing can generate depth d+1 and yields O(b^(d+1)); specify the convention. Explicit finite graph traversal with visited-state tracking takes O(V+E) time and O(V) space.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Question sources

Artificial Intelligence review questions · Question 45

Question 220 | NS0046

Representation learning in Deep Learning refers to:

A. Manually engineering input features

B. The ability to automatically discover and learn relevant features from raw data

C. Storing knowledge in frames and ontologies

D. Using labeled data with a teacher signal

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Automatically learning useful features from data.

AI34 says deep models learn features from images, sound and text through multiple representation layers. That is representation learning rather than manually specifying every input feature. A label signal may guide training but is not required for every deep objective, and frames/ontologies are separate explicit knowledge structures rather than the learned representations meant here.

Why each choice is right or wrong

A - Incorrect: Manual feature engineering is contrasted with learning the representations automatically.

B - Correct: The model learns task-useful features or representations from the supplied data.

C - Incorrect: Frames and ontologies are explicit knowledge representations, not the deep feature-learning mechanism described.

D - Incorrect: Teacher labels define supervision; representation learning can also use unsupervised or reinforcement objectives.

Primary note proof

Study-note section AI34

Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Study-note section AI43

ANNs, genetic search and reinforcement learning all improve candidate behavior or performance through experience in different ways. Deep learning is an architecture family usable with supervised, unsupervised or reinforcement objectives. Decision trees and deep models may both be supervised. To distinguish learning regimes, specify labels, reward or unlabeled structure.

Study-note section AI24

A semantic network connects concept nodes by labeled relations such as is-a. Frames describe stereotyped entities through slots, values and defaults. Production systems use IF conditions THEN conclusions or actions, matching rules against working memory and selecting among applicable rules. A graph suits relationships such as frequently bought together; frames suit attributes. No representation is always best for recommendation without task requirements.

Study-note section AI-S06

An ontology describes entities or individuals, classes and properties and may state logical relationships and restrictions. It organizes explicit conceptual knowledge. Conditional probability tables and a probabilistic factorization are additional modeling commitments, not inherent in the word ontology.

Question sources

Artificial Intelligence review questions · Question 46

Question 221 | NS0047

What makes Best-First (Greedy) Search non-optimal?

A. It always explores all possible paths

B. It relies entirely on heuristics and can overlook cheaper paths, making suboptimal choices

C. It uses actual path costs rather than estimates

D. It explores nodes in breadth-first order

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: It uses only h and can ignore a cheaper route with a larger estimate.

AI17 defines greedy best-first as f=h, ignoring cost already paid. AI19 demonstrates a route of cost 11 selected greedily while another route costs 4. The heuristic ranking can therefore prefer a worse total solution. Greedy is neither exhaustive nor breadth-first and does not order primarily by actual path cost.

Why each choice is right or wrong

A - Incorrect: Greedy search may stop at an early found goal rather than exploring every path.

B - Correct: Ignoring accumulated cost can favor a seemingly close but expensive route over a cheaper solution.

C - Incorrect: Using actual g alone describes UCS, whereas greedy uses h.

D - Incorrect: Breadth-first depth order is not the greedy minimum-heuristic policy.

Primary note proof

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI19

> For S→A cost 2, S→B cost 1, A→G cost 2, B→G cost 10, set h(A)=2, h(B)=1, h(G)=0. After expanding S, greedy chooses B because 1<2 and can then choose G, returning cost 11. A* compares A:f=2+2=4 and B:f=1+1=2; after B, its G entry costs 11. It next expands A and replaces G with cost 4, which is removed and accepted.

Question sources

Artificial Intelligence review questions · Question 47

Question 222 | NS0048

In AI, what does 'Acting Humanly' (Turing Test concept) primarily evaluate?

A. Whether a machine can solve math problems

B. Whether a machine's behavior is indistinguishable from a human's in conversation

C. Whether a machine can perceive visual stimuli

D. Whether a machine follows logical rules perfectly

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer B: Humanlike conversational behavior judged by indistinguishability.

AI02 asks whether a judge can reliably distinguish a machine from a human through dialogue. This operational test concerns observed conversation, not a proof of consciousness or perfect logical reasoning. Mathematical success and perception may contribute capabilities but are not the primary distinguishing conversational criterion. B states that criterion directly.

Why each choice is right or wrong

A - Incorrect: Mathematical ability alone does not supply the test's conversational indistinguishability criterion.

B - Correct: The test evaluates whether conversational behavior can be reliably distinguished from a human's.

C - Incorrect: Visual perception alone is not the conversational test described in AI02.

D - Incorrect: Perfect logic is a rationality criterion, not the test's humanlike observed-behavior requirement.

Primary note proof

Study-note section AI02

> Thinking humanly models human mental processes; thinking rationally derives conclusions by valid logic. Acting humanly seeks humanlike behavior; acting rationally selects actions expected to achieve objectives. Thought processes and observed behavior are different evaluation targets.

Study-note section AI02

> The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Question sources

Artificial Intelligence review questions · Question 48

Question 223 | NS0049

Which of the following is the correct relationship between AI, ML, and DL?

A. DL ⊂ AI ⊂ ML

B. AI ⊂ ML ⊂ DL

C. ML ⊂ DL ⊂ AI

D. DL ⊂ ML ⊂ AI

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer D: DL⊂ML⊂AI.

AI34 explicitly gives deep learning inside machine learning inside AI. Deep neural architectures are a subset of learning methods, and learning is one component of the broader intelligence field, which also includes planning and reasoning. Every other ordering reverses at least one of these stated inclusions.

Why each choice is right or wrong

A - Incorrect: This wrongly puts AI inside machine learning rather than the reverse.

B - Incorrect: Both inclusions are reversed relative to the note's hierarchy.

C - Incorrect: Machine learning contains deep learning, not the reverse.

D - Correct: Deep learning is contained in machine learning, which is contained in AI.

Primary note proof

Study-note section AI34

> Feedforward networks pass signals toward outputs without recurrent state loops; recurrent networks carry state across sequence steps. Training adjusts weights, often through gradients and backpropagation. Backpropagation is not the same as a recurrent connection. Deep learning uses multiple representation layers and can learn features from images, sound and text. CNNs share filters across image locations. ANN is a brain-inspired abstraction, not a literal brain simulation; DL⊂ML⊂AI.

Study-note section AI33

> An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Question sources

Artificial Intelligence review questions · Question 49

Question 224 | NS0050

A recommendation system must directly represent products and labeled “frequently bought together” relationships, then traverse those relationships to find connected products. Which listed representation most directly fits that requirement?

A. Rule-based IF-THEN logic

B. Frames describing product attributes

C. Graph-based model with products as nodes and relationships as edges

D. Probabilistic Bayesian network of user demographics

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Graph-based model with products as nodes and relationships as edges

A semantic graph represents entities as nodes and their labeled relationships as edges, making connected products directly traversable. Frames primarily organize attributes, rules express conditions and conclusions, and a Bayesian demographic model represents uncertain dependencies rather than the requested product-edge structure. The answer follows the explicit relational representation requirement; it does not claim that graphs are universally best for every recommendation system.

Why each choice is right or wrong

A - Incorrect: Rules can encode recommendations, but do not directly provide the requested product-node relationship structure.

B - Incorrect: Frames describe product attributes rather than making product-to-product links the primary structure.

C - Correct: Products as nodes and labeled relationships as edges directly support the required traversal.

D - Incorrect: A demographic Bayesian network models uncertain dependencies rather than the specified product-edge structure.

Primary note proof

Study-note section AI24

> A semantic network connects concept nodes by labeled relations such as is-a. Frames describe stereotyped entities through slots, values and defaults. Production systems use IF conditions THEN conclusions or actions, matching rules against working memory and selecting among applicable rules. A graph suits relationships such as frequently bought together; frames suit attributes. No representation is always best for recommendation without task requirements.

Study-note section AI29

> A Bayesian network is a directed acyclic graph with conditional probability tables; its joint distribution factors as the product of each node’s probability given its parents. A Markov model

assumes the next state depends on the current state rather than the entire past. These represent uncertainty; fuzzy membership represents a different issue.

Question sources

Artificial Intelligence review questions · Question 50

Question 225 | NS0156

What is the difference between supervised and unsupervised learning?

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer: Supervised learning fits labeled input-target examples; unsupervised learning discovers structure without target labels.

AI31 defines supervised learning through labels, with classification for discrete targets and regression for numeric targets. AI32 defines unsupervised discovery, including clustering, association and dimensionality reduction. The distinction is the supplied training feedback, not whether the data are images, text or numbers; the same modality can support either regime.

Primary note proof

Study-note section AI31

Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Question sources

Artificial Intelligence review questions · Question 1

Question 226 | NS0157

Under what search and heuristic conditions does A* find a least-cost path, and why?

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer: A* is optimal under its heuristic and search assumptions: admissible tree search, or consistent graph search without reopening; otherwise reopen improved states.

With admissible h, A* tree search uses f=g+h as a lower bound on solution cost through each frontier node. Removing a minimum-f goal cannot skip a cheaper represented solution. Graph search that never reopens closed states normally requires consistency; with an admissible inconsistent heuristic, improved states must be reopened. Goal testing is on removal. Finite branching and positive lower-bounded step costs supply the usual infinite-space completeness conditions.

Primary note proof

Study-note section AI18

> A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI18

> Admissibility suffices for optimal A* tree search; graph search that never reopens CLOSED states normally requires consistency. With an admissible inconsistent heuristic, reopen states when g improves. Test a goal when removed at minimum f. Completeness also needs finite branching and positive lower-bounded costs in infinite spaces. Worst-case time and memory can be exponential; no heuristic universally guarantees speed.

Question sources

Artificial Intelligence review questions · Question 2

Question 227 | NS0158

What is the Turing Test and what does it measure?

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer: The conversational Turing test asks a judge to distinguish machine from human through dialogue; it measures humanlike behavior, not consciousness.

AI02 sets the observable criterion: whether a judge can reliably tell a machine's conversation from a human's. Language use, knowledge, reasoning and often learning support the performance. Passing would meet this operational behavioral standard, but it does not prove consciousness, actual understanding or that the machine internally thinks in the same way as a human.

Primary note proof

Study-note section AI02

> The conversational Turing test asks whether a judge can reliably distinguish a machine from a human through their dialogue. It requires language use, knowledge, reasoning and often learning. It tests an operational behavior criterion, not proof of consciousness. Cognitive models can be tested against response times, errors and psychological experiments.

Question sources

Artificial Intelligence review questions · Question 3

Question 228 | NS0159

What is the difference between precision and recall?

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer: Precision=TP/(TP+FP): how many predicted positives are correct. Recall=TP/(TP+FN): how many actual positives are found.

AI36 distinguishes precision's predicted-positive denominator from recall's actual-positive denominator. False positives reduce precision; false negatives reduce recall. With TP=30, FP=10 and FN=20, precision is 30/40=75% and recall is 30/50=60%. The positive class and zero-denominator convention should be fixed before comparing scores.

Primary note proof

Study-note section AI36

> With a fixed positive class, TP means true positives, FP false positive predictions, FN missed positives, TN true negatives. Accuracy=(TP+TN)/N; precision=TP/(TP+FP); recall=TP/(TP+FN); F1=2PR/(P+R). State the averaging and zero-denominator convention when relevant.

Question sources

Artificial Intelligence review questions · Question 4

Question 229 | NS0160

Why is tokenization important in natural language processing?

Hint

Match the input, output and purpose to the appropriate language, perception or physical-action task.

Answer: Tokenization divides text into processing units such as words, punctuation or subwords so later NLP stages can operate on a consistent input sequence.

AI37 defines tokenization as dividing text into processing units and lists several permitted granularities. Establishing that sequence lets later tagging, recognition and parsing refer to units consistently. Tokens need not equal whitespace-separated words, and tokenization by itself does not determine grammatical structure or resolve word meaning; those belong to later analyses.

Primary note proof

Study-note section AI37

> NLP interprets and generates human language: translation, dialogue, summarization, question answering and text generation. Speech recognition converts audio to text; speech synthesis generates speech. They overlap language processing but solve different input/output tasks.

Study-note section AI37

> Tokenization divides text into processing units; tokens may be words, punctuation or subwords. POS tagging assigns grammatical roles; named-entity recognition identifies people, organizations and locations. Parsing identifies grammatical structure; semantics concerns meaning; discourse links sentences; pragmatics uses context and speaker intent. Sentiment analysis predicts evaluative tone. Ambiguity can occur at every level: “bank” has multiple senses, and “I saw the man with a telescope” has multiple attachments.

Question sources

Artificial Intelligence review questions · Question 5

Question 230 | NS0285

Which AI approach most directly supports understanding biological intelligence behavior as a scientific goal?

A. Weak AI

B. Applied AI

C. Cognitive AI

D. Strong AI

Hint

Read the definition and the classification axis requested; distinguish the defining purpose from supporting techniques or implementation resources.

Answer C: Cognitive AI.

AI01 explicitly describes cognitive AI as using computational models to investigate biological intelligence, making it the direct scientific-goal match. Applied AI emphasizes useful solutions. Narrow AI describes capability scope, while strong AI has a philosophical understanding sense; neither is the named research approach for testing models against biological behavior.

Why each choice is right or wrong

A - Incorrect: Weak AI denotes bounded capability rather than the specific scientific modeling goal.

B - Incorrect: Applied AI builds useful solutions rather than primarily explaining biological intelligence.

C - Correct: Cognitive AI uses computational models to investigate biological intelligence and behavior.

D - Incorrect: Strong AI concerns broad or actual-understanding claims, not this specific empirical modeling approach.

Primary note proof

Study-note section AI01

> Artificial intelligence studies systems that perform tasks requiring intelligence: reasoning, learning, planning, perception and language. Applied AI builds useful solutions; cognitive AI uses computational models to investigate biological intelligence. A model should be compared with human behavior or experimental observations before claiming it explains human thought.

Study-note section AI01

> Narrow or weak AI addresses bounded tasks, such as face recognition, spam filtering or a voice assistant. General AI denotes broad competence transferable across domains; strong AI also has a philosophical meaning involving actual understanding. Do not equate a humanoid body, autonomous vehicle, or many features with general intelligence.

Question sources

Computer Science Exit Exam · 2018 · Question 2

Question 231 | NS0291

Why is Greedy Best-First Search not guaranteed to be optimal or complete?

A. It ignores path cost g(n) and relies only on h(n)

B. It always chooses the shallowest node

C. It expands nodes in FIFO order

D. It requires full knowledge of the goal state in advance

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: It uses h alone, ignoring accumulated g.

AI17 says greedy best-first chooses f=h and can take an expensive route or follow an infinite misleading branch. Ignoring g explains nonoptimality, while cycles or infinite paths without appropriate handling explain failure of completeness. On a finite graph with full duplicate handling it is complete, so the lack of a general guarantee does not mean every implementation fails.

Why each choice is right or wrong

A - Correct: Using only h ignores total path cost and can follow misleading or unbounded branches.

B - Incorrect: Shallowest-first describes breadth-first order rather than greedy heuristic order.

C - Incorrect: FIFO is breadth-first queue behavior, not minimum-h priority selection.

D - Incorrect: Heuristics estimate promise; exact full goal-state knowledge is not required by the algorithm.

Primary note proof

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI19

> For S→A cost 2, S→B cost 1, A→G cost 2, B→G cost 10, set h(A)=2, h(B)=1, h(G)=0. After expanding S, greedy chooses B because 1<2 and can then choose G, returning cost 11. A* compares A:f=2+2=4 and B:f=1+1=2; after B, its G entry costs 11. It next expands A and replaces G with cost 4, which is removed and accepted.

Question sources

Computer Science Exit Exam · 2018 · Question 8

Question 232 | NS0305

Which search strategy tries to expand the node that is closest to the goal, on the grounds that this is likely to lead to a solution quickly; Thus, it evaluates nodes by using just the heuristic function: f (n) = h (n).

A. Uniform-cost search

B. Greedy best-first search

C. Depth-first search

D. A* search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer B: Greedy best-first search.

AI17 explicitly defines greedy best-first by removing the smallest remaining-cost estimate h while ignoring paid cost g. This is exactly the stem's equation. UCS uses g, A* combines g and h, and DFS uses depth/stack discipline rather than a minimum heuristic evaluation.

Why each choice is right or wrong

A - Incorrect: UCS orders by accumulated g rather than h alone.

B - Correct: Greedy best-first selects the node with the smallest heuristic estimate h.

C - Incorrect: DFS follows a LIFO/deepest-branch order rather than minimum h.

D - Incorrect: A* uses g+h, while the stem explicitly uses only h.

Primary note proof

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI15

> UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Study-note section AI18

> A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Question sources

Computer Science Exit Exam · 2018 · Question 23

Question 233 | NS0312

Which of the following best defines a heuristic in the context of AI search?

A. A function that estimates how close a state is to the goal

B. A guaranteed optimal path to the goal

C. A method that always avoids revisiting states

D. A brute-force enumeration of all possible states

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: A problem-specific estimate of remaining cost or promise.

AI17 defines a heuristic h(n) as estimating remaining cost or evaluating promise. A's closeness wording captures that estimate rather than an exact path guarantee. A heuristic does not inherently prevent repeated states and is not exhaustive enumeration; those are separate search-policy or bookkeeping choices. Its quality and computation cost affect practical search.

Why each choice is right or wrong

A - Correct: Estimating closeness or remaining cost is the heuristic role defined in AI17.

B - Incorrect: An estimate is not itself a guaranteed optimal path.

C - Incorrect: Repeated-state avoidance requires duplicate handling rather than following from heuristic estimation.

D - Incorrect: Exhaustive enumeration is a search method, not the definition of a heuristic estimate.

Primary note proof

Study-note section AI17

> A heuristic h(n) estimates remaining cost or evaluates promise using problem-specific knowledge. It can reduce explored work, but computing it also costs time. Uninformed algorithms use only the problem definition and known costs; UCS remains uninformed despite using g.

Study-note section AI12

> A search node records a state plus parent, action, depth and path cost. Expanding a node generates its successors. OPEN or frontier stores discovered candidates awaiting expansion; CLOSED records expanded states. Reaching the same state through different paths does not imply the same node or cost.

Question sources

Computer Science Exit Exam · 2018 · Question 35

Question 234 | NS0326

Which type of heuristic search strategy that evaluates nodes by combining g (n), the cost to reach the node, and h (n.), the cost to get from the node to the goal: f (n) = g (n) + h (n) Since g(n) gives the path cost from the start node to node n, and h(n) is the estimated cost of the cheapest path from n to the goal, we have f (n) = estimated cost of the cheapest solution through n ?

A. A* search

B. Greedy Best-First Search

C. Uniform-cost search

D. Depth-first search

Hint

Check how the method selects its next candidate, what information it uses, and which cost, depth or heuristic assumptions the question supplies.

Answer A: A* search.

AI18 defines A* by f(n)=g(n)+h(n), exactly as the long stem specifies. The sum estimates the cheapest solution through the node when h estimates remaining cost. Greedy ignores g, UCS ignores h, and DFS does not prioritize this sum, so none of those matches the stated evaluation.

Why each choice is right or wrong

A - Correct: A* adds actual cost-to-node g and estimated remaining cost h.

B - Incorrect: Greedy best-first uses only h and omits accumulated path cost.

C - Incorrect: UCS uses only g and does not add a remaining-cost heuristic.

D - Incorrect: DFS uses stack/depth ordering rather than minimum g+h.

Primary note proof

Study-note section AI18

> A* orders by f(n)=g(n)+h(n). Admissibility means 0≤h(n)≤h*(n), the true cheapest remaining cost. Consistency means h(n)≤c(n,n′)+h(n′) on every edge, with h(goal)=0; it implies admissibility on states with goal paths. Consistency makes f nondecreasing along a path.

Study-note section AI17

> Best-first is a family that orders OPEN by an evaluation function. Greedy best-first uses f=h and removes the smallest estimate, ignoring the cost already paid. It can choose an expensive route, loop without duplicate checks, or follow an infinite misleading branch. On a finite graph with full duplicate handling it is complete, but not generally optimal. Its implicit finite-tree worst case can be O(b^m).

Study-note section AI15

UCS uses a priority queue ordered by actual accumulated path cost g(n). Remove a minimum-g node; accept a goal when removed, not when first generated. Relax each edge: if a cheaper path is found, update its frontier cost and parent.

Study-note section AI14

DFS follows the deepest pending branch using a LIFO stack or recursion. FILO means the same order as LIFO. With the same children as BFS, a left-first DFS visits S,A,C before B; push rightmost children first to pop the leftmost first.

Question sources

Computer Science Exit Exam · 2018 · Question 52

Question 235 | NS0349

In which of the following Learning methods an agent receives feedback in the form of rewards or penalties to maximize cumulative reward?

A. Supervised learning

B. Reinforcement learning

C. Semi-supervised learning

D. Unsupervised learning

Hint

Identify the training feedback, prediction target or measured error. Keep learning objectives separate from model architecture and data modality.

Answer B: Reinforcement learning.

AI33 defines RL through interaction, rewards and improvement of a policy to maximize expected cumulative reward. The reward may be delayed and does not provide a correct-action label. Supervised and semi-supervised learning instead concern target labels, while unsupervised discovery lacks the defining action-reward interaction described here.

Why each choice is right or wrong

A - Incorrect: Supervised learning fits labeled targets rather than defining behavior through cumulative interaction rewards.

B - Correct: Reward or penalty feedback guiding cumulative-return improvement is the defining RL mechanism.

C - Incorrect: Semi-supervised learning mixes labeled and unlabeled examples, not necessarily action rewards.

D - Incorrect: Unsupervised learning discovers unlabeled structure rather than optimizing an interaction reward policy.

Primary note proof

Study-note section AI33

An RL agent observes state information, takes an action, receives reward and a new observation, and improves its policy to maximize expected cumulative reward. Rewards evaluate outcomes rather than supply the correct action label. Rewards may be delayed; not every correct-looking action receives immediate reward.

Study-note section AI31

Supervised learning fits a mapping from labeled input-target examples. Classification predicts discrete labels, such as institution generations or spam/nonspam; regression predicts numeric quantities, such as prices. Predictive analytics uses historical patterns to forecast outcomes and can use either. Specify the target before choosing a method.

Study-note section AI32

Semi-supervised learning combines some labels with additional unlabeled data. Active learning chooses informative cases for an oracle to label; it still uses labels. “Exploratory learning” is not an exclusive standard synonym: it can mean unsupervised exploration or active data acquisition, so define the feedback. A task named summarization or time-series analysis is not automatically supervised; its training targets determine that.

Study-note section AI32

Unsupervised learning seeks structure without target labels: clustering groups similar examples; association finds co-occurrence; dimensionality reduction compresses features. Good clusters have high within-cluster similarity and low between-cluster similarity under an appropriate distance. K-means alternates assignment to nearest centroids and centroid updates.

Question sources

Computer Science Exit Exam · 2018 · Question 83

Computer Security

Questions & Worked Answers

171 questions

Question 1 | Q00002

Which one of the following is an attack performed by wiretapping a network and illicitly copy files and programs?

A. Interception

B. Fabrication

C. Interruption

D. Modification

Hint

Classify the action actually described and distinguish observation, alteration, disruption and identity claims.

Answer A: Interception

Wiretapping and copying give an unauthorized party access without changing or destroying the resource. This is the disclosure/interception case on PDF p.4.

Why each choice is right or wrong

A - Correct: Interception directly matches unauthorized snooping/access. See PDF p.4.

B - Incorrect: Fabrication inserts a fake resource; no insertion is described. See PDF p.4.

C - Incorrect: Interruption makes the resource unavailable; copying does not imply loss of service. See PDF p.4.

D - Incorrect: Modification changes the resource; no change is stated. See PDF p.4.

Primary note proof

Study-note section SEC03

> Interception copies or observes data, modification alters it, interruption blocks it, and fabrication inserts false material. The 4D classification uses disclosure, deception, disruption and usurpation, the last meaning unauthorized control. DoS primarily demonstrates disruption/availability loss; it need not give control of the victim.

Study-note section SEC03

> Passive attacks observe without changing the target exchange: sniffing and traffic analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 50-61

• Types of Attacks on Data CIA (W. Stallings)
o Disclosure/Interception: unauthorized party snooping, or getting access to a resource, Attack on data confidentiality:
o Modification / Deception: unauthorized party modifying a resource, acceptance of false data, E.g., providing wrong data (attack on data integrity)
o Disruption/Interruption: destroyed/unavailable services/resources, DoS (attack on data availability)
o Fabrication: unauthorized party inserts a fake asset/resource, loss of confidentiality, authenticity and integrity of the message

Question sources

MOE Computer Science Exit Exam · 2016 · Question 2

Question 2 | Q00004

Assuming the UPS is correctly installed and operating normally, which listed item is a protective power-supply control rather than an environmental hazard?

A. Improperly placed heater

B. Humidity

C. Lightning strikes

D. UPS(uninterruptible power supply)

Hint

Start with the stated attacker path or physical hazard and the limits of each control.

Answer D: UPS(uninterruptible power supply)

A working UPS supplies temporary conditioned power and normally protects equipment against power problems. The other entries name heat-placement, humidity or electrical-surge hazards that can impair operation. The revised question assumes normal installation and operation only for the UPS, rather than claiming it can never cause harm. Battery faults and installation failures remain possible in general, but they are outside this stated comparison.

Why each choice is right or wrong

A - Incorrect: An improperly placed heater can expose equipment to damaging heat.

B - Incorrect: Humidity is an environmental condition requiring protection.

C - Incorrect: Lightning can produce damaging electrical surges.

D - Correct: A correctly installed, working UPS supplies protective conditioned power.

Primary note proof

Study-note section SEC11

Protect equipment against theft, unauthorized connections, heat, humidity, water, electrical surges and fire. A UPS supplies temporary conditioned power; it is normally protective, but faulty installation/battery failure can itself be hazardous. A disconnected machine loses many network attack paths yet remains exposed to insiders, removable media, supply-chain attacks and physical access. There is no universal “most secure computer” without a threat model.

Study-note section SEC11

Choose protection proportional to asset value and credible threats.

Study-note section SEC11

Uniform configurations simplify patching and prediction but create correlated failures; diversity may reduce common-mode compromise while raising administration cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 65-68

4. Physical security
• Without physical security to protect hardware (i.e., doors that lock) nothing else about a computer system can be called secure

Computer Security module | PDF page 4 | lines 89-99

o Encryption: primary control that protects CIA, by which Cleartext scambled into ciphertext.
o Software controls: password checker, virus scanner, IDS (intrusion detection system)
o Hardware controls: provide higher degree of security; Locks, Smart cards, dongles, hadware keys...
o Policies and procedures: Alignment with users’ legal and ethical standards; eg. password policy
o Physical controls: natural and man-made disaster protection; walls, locks, guards, security, cameras, backup copies and archives

Question sources

MOE Computer Science Exit Exam · 2016 · Question 4

Question 3 | Q00006

Awgichew wants message confidentiality for Meron using asymmetric encryption, not a signature. KPRIV-A/KPUB-A are Awgichew's keys and KPRIV-M/KPUB-M are Meron's keys. Which statements are NOT correct for that confidential exchange? Select all that apply.

A. Awgichew uses Meron's private key KPRIV-M to encrypt his plain text

B. Awgichew uses his KPUB-A to encrypt his plain text

C. Meron can decode the encrypted text using her KPRIV-M

D. Meron can decode the scrambled text using her KPUB-M

Hint

Identify the intended recipient or signer and separate confidentiality from verification.

Answer A, B, D: Awgichew uses Meron's private key KPRIV-M to encrypt his plain text; Awgichew uses his KPUB-A to encrypt his plain text; Meron can decode the scrambled text using her KPUB-M

For confidentiality to Meron, the sender encrypts with Meron's authenticated public encryption key and she decrypts with her private key. Awgichew should neither possess her private key nor encrypt to his own public key for a message only she should read. Her public key is not the secret decryption key. Thus A, B and D are incorrect directions; C describes the valid recipient decryption step and is not selected in this NOT-correct question.

Why each choice is right or wrong

A - Correct: Incorrect exchange step: the sender must not use or assume possession of Meron's private key.

B - Correct: Incorrect exchange step: encrypting to Awgichew's public key addresses the wrong recipient.

C - Incorrect: This is a valid decryption step, so it must not be selected as NOT correct.

D - Correct: Incorrect exchange step: Meron decrypts with her private key, not her public key.

Primary note proof

Study-note section SEC18

> For confidentiality to B, encrypt with B's authenticated public encryption key; B decrypts with B's private key. Do not encrypt to A's public key if only B should read it, and never assume A possesses B's private key. For a signature, A signs with A's private signing key and others verify with A's public key. These are distinct operations; not every public-key algorithm supports encryption.

Study-note section SEC18

> For one-way public-key confidentiality the sender need not own a key pair; it needs the recipient's authenticated public encryption key.

Study-note section SEC18

A sender key pair is needed when the protocol requires its signature or separate recipient role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 8 | lines 61-76

Asymmetric key cryptography
• also known as public-key encryption.
• In public key cryptography,
o unlike symmetric-key, there are two keys: a private key and a public key.
✓ The public key is announced to the public; whereas the private key is kept by the receiver.
o anyone can send messages using the public key of the receiver for encryption, but only the receiver uses his private key for decryption
✓ Secret transmission of key for decryption is not required
✓ Every entity can generate a key pair and release its public key
✓ Encryption and decryption keys are different but form a unique pair

Computer Security module | PDF page 8 | lines 89-98

• Asymmetric-key cryptosystem Model
o Both users (Bob & Alice) generates a pair of keys
o Each user places one of 2 keys in public register.
✓ This is the public key: each user maintains a collection of public keys obtained from others.
o The companion key is kept private to each.
o If Alice wishes to send a confidential message to Bob, She encrypts the message using Bob's public key.
o When Bob receives the message, he decrypts it using his private key.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 6

Question 4 | Q00009

In a two-way confidential exchange, both participants send and receive encrypted messages. Which statement correctly describes symmetric versus asymmetric encryption key arrangements?

A. Both use the same shared secret as their sole confidentiality key

B. Symmetric key requires the sender and receiver to have a public and private key

C. Asymmetric encryption requires each participant, acting as a recipient, to have its own public/private encryption key pair

D. Both are recovery methods

Hint

Identify the intended recipient or signer and separate confidentiality from verification.

Answer C: Asymmetric encryption requires each participant, acting as a recipient, to have its own public/private encryption key pair

Symmetric encryption uses a secret shared by the communicating parties. For public-key confidentiality, a sender encrypts to the recipient's authenticated public encryption key, and that recipient decrypts with its private key. Because both participants receive encrypted messages in this two-way scenario, each needs its own encryption key pair. A sender in a strictly one-way exchange would not need a pair merely to send. Neither architecture is defined as a recovery method.

Why each choice is right or wrong

A - Incorrect: False: a shared secret describes symmetric confidentiality, not the asymmetric key arrangement.

B - Incorrect: False: symmetric encryption does not require each party to have a public/private pair.

C - Correct: Correct: each participant is a recipient in one direction and needs its own public/private encryption pair.

D - Incorrect: False: encryption key arrangements protect communication; they are not recovery methods.

Primary note proof

Study-note section SEC18

> For one-way public-key confidentiality the sender need not own a key pair; it needs the recipient’s authenticated public encryption key. A sender key pair is needed when the protocol requires its signature or separate recipient role.

Study-note section SEC16

> Symmetric encryption uses a shared secret and efficiently protects bulk data.

Study-note section SEC18

For confidentiality to B, encrypt with B's authenticated public encryption key; B decrypts with B's private key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 7 | lines 48-54

Symmetric key cryptography
• Uses conventional / secret-key / single-key.
• The secret key shared by both sender and receiver.
• All classical encryption algorithms are private-key.
o Use the same key (shared key) for encryption and decryption process.
• If this key is disclosed communications are compromised.

Computer Security module | PDF page 8 | lines 61-76

Asymmetric key cryptography
• also known as public-key encryption.
• In public key cryptography,
o unlike symmetric-key, there are two keys: a private key and a public key.
✓ The public key is announced to the public; whereas the private key is kept by the receiver.
o anyone can send messages using the public key of the receiver for encryption, but only the receiver uses his private key for decryption
✓ Secret transmission of key for decryption is not required
✓ Every entity can generate a key pair and release its public key
✓ Encryption and decryption keys are different but form a unique pair

Question sources

MOE Computer Science Exit Exam · 2016 · Question 9

Question 5 | Q00030

A cipher replaces each plaintext letter with the next letter in the alphabet, wrapping Z to A. Which is the most specific standard cipher classification of this operation?

A. Transliteration

B. Translation

C. Transformation

D. Substitution

Hint

Separate symbol replacement from position changes and apply the alphabet and modular direction.

Answer D: Substitution

The operation changes each symbol's identity while leaving its position in the message unchanged. That is substitution, and using the next alphabet letter is a fixed Caesar shift. General text-conversion descriptions do not identify this specific cipher class. Transformation is especially broad and could include many cryptographic or noncryptographic changes. D is therefore the most precise classification, rather than a claim that ordinary language can never describe encryption using broader terms.

Why each choice is right or wrong

A - Incorrect: Incorrect: transliteration does not identify the specified fixed-alphabet-shift cipher class.

B - Incorrect: Incorrect: translation does not identify the specific cipher class defined by replacing symbols while retaining positions.

C - Incorrect: Incorrect: transformation is a broader description and does not distinguish substitution from other changes.

D - Correct: Correct: replacing symbol identities while keeping their positions is substitution.

Primary note proof

Study-note section SEC13

> Substitution replaces symbols; transposition changes their positions. A monoalphabetic substitution preserves one mapping throughout the message and leaks frequency patterns despite a large nominal key space. Classical ciphers are teaching devices, not modern data protection.

Study-note section SEC13

> With A=0 through Z=25, Caesar encryption is C=(P+k) mod26 and decryption P=(C-k) mod26. A shift of 3 maps A→D and HELLO→KHOOR.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 19-28

> Substitution ciphers
> • Replaces one symbol with another.
> • Categories monoalphabetic or polyalphabetic ciphers.
> (1) Monoalphabetic Ciphers
> o The relationship between a symbols in the plaintext to a
> symbol in the ciphertext is always one-to-one.
> o sometimes called additive cipher or shift cipher or Caesar
> cipher (key of 3)

o When the cipher is additive, the plaintext, ciphertext, and key are integers in Z26.

Computer Security module | PDF page 6 | lines 73-78

Transposition Ciphers
• Does not substitute one symbol for another, instead it changes the location of the symbols (reorders symbols).
• A transposition cipher relies on an algorithm which rearranges the order of the letters in a plaintext message

Question sources

MOE Computer Science Exit Exam · 2016 · Question 30

Question 6 | Q00039

Which one of the following is an attack detection mechanism?

A. Physical control

B. Audit log

C. Password

D. Encryption

Hint

Distinguish weaknesses, potential harmful events, actual attempts and the effect of a control.

Answer B: Audit log

Logs preserve successful and failed authentication events (PDF p.4). Reviewing this evidence can determine that an attack occurred, matching the module's detection objective; the other listed controls primarily restrict or protect.

Why each choice is right or wrong

A - Incorrect: Physical controls protect facilities; this option does not specify monitoring evidence. See PDF p.4.

B - Correct: An audit log records relevant events for identifying attacks after or during occurrence. See PDF p.4.

C - Incorrect: A password checker controls entry rather than recording evidence of attack by itself. See PDF p.4.

D - Incorrect: Encryption scrambles cleartext; it does not by itself identify an attack. See PDF p.4.

Primary note proof

Study-note section SEC02

Preventive controls block events, detective controls reveal them, corrective/recovery controls restore safe operation, and deterrents discourage attempts. Encryption is preventive for disclosure; an audit log is principally detective/accountability evidence. Physical, technical and administrative are control categories, not competing stages. An external firewall cannot replace a written policy or a locked equipment room.

Study-note section SEC01

Authentication establishes an identity or other claim; authorization decides permitted actions.

Study-note section SEC01

Accountability links actions to responsible entities through reliable identity and protected logs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 28-35

• Accountability
o Ensuring an entity's action is traceable uniquely to that entity.
o Ascertains the responsibility of an entity (like a person) for its actions and decisions.
o All relevant activities events and operations on a system, e.g., failed and successful authentication attempts, are recorded in a log.

Computer Security module | PDF page 4 | lines 82-84

o Detect attack: accept that an attack will occur; the goal is to determine that an attack is under way, or has occurred, and report it.

Computer Security module | PDF page 4 | lines 89-99

o Encryption: primary control that protects CIA, by which Cleartext scambled into ciphertext.
o Software controls: password checker, virus scanner, IDS (intrusion detection system)
o Hardware controls: provide higher degree of security; Locks, Smart cards, dongles, hadware keys...
o Policies and procedures: Alignment with users' legal and ethical standards; eg. password policy
o Physical controls: natural and man-made disaster protection; walls, locks, guards, security, cameras, backup copies and archives

Question sources

MOE Computer Science Exit Exam · 2016 · Question 39

Question 7 | Q00048

Which one of the following best describes for a program that seems friendly and legitimate but contains hidden malicious codes and harms a system?

A. Trojan

B. Malware

C. Virus

D. Worm

Hint

Separate propagation, disguise and payload when classifying malicious software.

Answer A: Trojan

The defining clue is apparent legitimacy hiding malicious behavior. SEC04 assigns that disguise property to a Trojan. Malware is the broad umbrella, while viruses and worms are distinguished by replication mechanisms. Because the stem asks what best describes the program, Trojan is more specific than the umbrella term and requires no unmentioned replication behavior.

Why each choice is right or wrong

A - Correct: A Trojan presents useful or legitimate behavior as cover for concealed malicious actions.

B - Incorrect: Malware is the broad umbrella and does not specifically capture the deceptive legitimate appearance asked for.

C - Incorrect: A virus is defined by attaching to a host and replicating, neither of which the stem specifies.

D - Incorrect: A worm independently replicates; that is different from the concealed malicious disguise described.

Primary note proof

Study-note section SEC04

> A virus replicates by attaching to a host program or document; execution of the infected host spreads it. A worm replicates independently, commonly across networks. A Trojan disguises malicious behavior as something useful and does not require self-replication. These labels describe different properties, so a malware package may contain several components.

Study-note section SEC04

> Malware is the umbrella term, not a specific replication method.

Study-note section SEC04

> Update, restrict execution, isolate compromise and maintain tested offline backups; no antivirus guarantees complete protection.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 21-35

• Malware Attack: A generic term for software that has
malicious purpose.
o Viruses
o Worms
o Trojan horses
o Spy-wares
o Zombie
o Botnets
o Bacterium
o Logic bomb
o Backdoor(Trapdoor)
o Adware
o Flooders (DoS client)
o Key loggers
o Rootkit

Question sources

MOE Computer Science Exit Exam · 2016 · Question 48

Question 8 | Q00055

A malicious program that once installed on your machine can collect all keystrokes including username, password, emails, credit card numbers and other sensitive information is:

A. Rootkit

B. Worms

C. Key logger

D. Trojan

Hint

Separate propagation, disguise and payload when classifying malicious software.

Answer C: Keylogger

SEC04 directly assigns keystroke recording to a keylogger. Recording every entered character can capture passwords and other secrets typed by the user. Rootkits hide malicious presence, worms replicate independently and Trojans conceal malicious behavior under useful appearances; none of those names specifically denotes keystroke collection.

Why each choice is right or wrong

A - Incorrect: A rootkit hides malicious presence or privileged activity; keystroke capture is not its defining function.

B - Incorrect: Worms are defined by independent replication, not by collecting typed information.

C - Correct: A keylogger records keystrokes and can capture the sensitive values entered through the keyboard.

D - Incorrect: A Trojan describes deceptive packaging; it does not specifically name the keystroke-recording function.

Primary note proof

Study-note section SEC04

> A virus replicates by attaching to a host program or document; execution of the infected host spreads it. A worm replicates independently, commonly across networks. A Trojan disguises malicious behavior as something useful and does not require self-replication. These labels describe different properties, so a malware package may contain several components.

Study-note section SEC04

> A backdoor bypasses ordinary access paths; a rootkit hides malicious presence or privileged activity.

Study-note section SEC04

> Spyware monitors users; a keylogger records keystrokes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 21-35

> • Malware Attack: A generic term for software that has
> malicious purpose.
> o Viruses
> o Worms
> o Trojan horses
> o Spy-wares
> o Zombie
> o Botnets
> o Bacterium
> o Logic bomb
> o Backdoor(Trapdoor)
> o Adware
> o Flooders (DoS client)
> o Key loggers
> o Rootkit

Computer Security module | PDF page 5 | lines 65-74

> • Reconnaissance: The hackers first identifies a target to
> launch an attack, extract maximum information regarding
> this target, understand its vulnerabilities, and then only
> explore the best ways to exploit it.

o Examples: Credential Theft using Keyloggers and Spyware; Identity Theft using spam e-mail, phishing attack and social engineering attack
• Access: Ones the attacker Reconnaissance the target, S/he starts a successful exploitation and continued access to the system. Then, the attacker is in control of the target.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 55

Question 9 | Q00187

Which authentication factor class uses a characteristic of the user's body, such as a fingerprint?

A. One-factor authentication

B. Two-factor authentication

C. Inherence (biometric) factor

D. Forth-factor authentication

Hint

Separate identity claims from proof, and factor type from factor count.

Answer C: Inherence (biometric) factor

An inherence factor authenticates using a biometric characteristic such as a fingerprint. That describes a factor class, not a count of independent classes combined by a procedure. One biometric alone is one factor, and biometrics may also participate in multi-factor authentication. The option identifies the factor class, while the numerical alternatives do not identify the bodily characteristic being requested.

Why each choice is right or wrong

A - Incorrect: One factor is a count and does not specify whether the factor is biometric.

B - Incorrect: Two-factor authentication is a combination count, not the body-based factor class.

C - Correct: Inherence identifies the biometric factor class.

D - Incorrect: Four-factor authentication is a count rather than the requested factor class.

Primary note proof

Study-note section SEC07

Knowledge is a password/PIN, possession is a token/card, and inherence is a biometric such as a fingerprint. Behavioral biometrics include typing rhythm or handwriting; some textbooks call these "something one does." A body characteristic is not "three-factor authentication": one biometric is one factor. Password plus PIN remains one factor class; card plus PIN is two. An OTP's factor depends on its delivery/generation mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 74-79

• Most OS authentication is handled through user names and passwords. Biometric (e.g. voice, face, retina, iris, fingerprint) and physical token-based (swipe cards, pin-generating cards) authentication are sometimes used to augment simple passwords, but the costs and accuracy of the technology limit their adoption.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Question sources

Addis Ababa University mock exam · Question 89

Question 10 | Q00190

is a system that monitors traffic into and out of a network and automatically alerts personnel when
suspicious traffic patterns occur, indicating a possible unauthorized intrusion attempt.

A. Anti-Virus Software

B. IDS

C. Firewall

D. Router

SS

Hint

Separate observation from blocking and sensor placement from detection method.

Answer B: Intrusion detection system

The stem describes observing traffic and alerting on suspicious patterns rather than enforcing a block. SEC24 defines that as an IDS role. Antivirus focuses on malicious software, a firewall enforces traffic policy and a router forwards packets. A product can combine functions, but detection/alerting is the specific function requested.

Why each choice is right or wrong

A - Incorrect: Antivirus targets malicious software; it is not the generic traffic-pattern detection system defined by the stem.

B - Correct: An IDS observes activity and alerts personnel about suspected intrusion patterns.

C - Incorrect: A firewall's defining role is traffic-policy enforcement rather than the stated intrusion-alerting function.

D - Incorrect: A router forwards packets between networks and is not defined by suspicious-pattern alerting.

Primary note proof

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline. Host-based sensors examine host events, network-based sensors inspect traffic. Signature detection recognizes known patterns; anomaly detection compares with a baseline and may detect novel behavior. Neither guarantees every attack is detected; anomaly detection can raise many false positives, and signatures can sometimes match variants.

Study-note section SEC-S03

> A router forwards IP packets between networks; forwarding alone is different from detecting suspicious behavior or enforcing a firewall's security policy.

Study-note section SEC23

> A firewall enforces traffic policy.

Related reading (context, not primary note proof)

Study-note section SEC04

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 89-99

> o Encryption: primary control that protects CIA, by
> which Cleartext scambled into ciphertext.
> o Software controls: password checker, virus scanner,
> IDS (intrusion detection system)
> o Hardware controls: provide higher degree of security;
> Locks, Smart cards, dongles, hadware keys...
> o Policies and procedures: Alignment with users' legal

and ethical standards; eg. password policy
o Physical controls: natural and man-made disaster protection; walls, locks, guards, security, cameras, backup copies and archives

Computer Security module | PDF page 4 | lines 82-84

o Detect attack: accept that an attack will occur; the goal is to determine that an attack is under way, or has occurred, and report it.

Computer Security module | PDF page 11 | lines 22-34

3. Firewall
• a program or network devices (e.g. router) which filters access to a protected network from the internet connection.
• protects a local or network of systems from network-based security threats,
• monitors & controls incoming & outgoing network traffic based on predetermined security rules.
• Firewall Objectives
o Keep intruders, malicious code and unwanted traffic out
o Keep private and sensitive information in
o security wall b/n private (protected) NW & outside word.

Question sources

Addis Ababa University mock exam · Question 92

Computer Science Exit Exam · Sene 2016 · Question 21-3

Question 11 | Q00231

Which attack category is defined by exhausting or disabling operating-system services so legitimate users cannot obtain service, regardless of the mechanism used?

A. Buffer Overflow

B. Malware

C. Denial of Service

D. Network Intrusion

Hint

Match the resource bottleneck or wireless mechanism to the relevant protection layer.

Answer C: Denial of Service

Denial of service is defined by exhausting or disabling a resource so legitimate users cannot use it. A buffer overflow is a memory-bounds violation that can have several consequences, including a crash in some cases; malware names malicious software, and network intrusion concerns unauthorized entry. Any of those mechanisms could participate in a service-denial incident, but they are not the availability-outcome category requested by the stem. C supplies that category.

Why each choice is right or wrong

A - Incorrect: Incorrect: a buffer overflow writes beyond allocated bounds; it is a possible mechanism rather than the service-denial category.

B - Incorrect: Incorrect: malware is an umbrella class of malicious software, not the requested availability outcome.

C - Correct: Correct: denial of service exhausts or disables a resource and blocks legitimate service.

D - Incorrect: Incorrect: network intrusion concerns unauthorized entry; the requested category is loss of service.

Primary note proof

Study-note section SEC06

> DoS exhausts or disables a resource; distributed DoS uses multiple sources, often bots or reflected/amplified responses.

Study-note section SEC25

> A buffer overflow writes beyond allocated bounds and can corrupt data or control information; code execution is a possible consequence, not its definition or an inevitable outcome.

Study-note section SEC04

> Malware is the umbrella term, not a specific replication method.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 42-47

> • Denial of Service (DoS) Attack: Blocking access from legitimate users
> • Distributed DoS Attack: is accomplished by tricking routers into attacking a target or using Zombie hosts to simultaneously attack a given target with large number of packets.

Question sources

Mock exam 4 · Question 33

Question 12 | Q00234

Which statement is false about network-traffic denial-of-service attacks? Treat NXDOMAIN flooding here as distributed and fragmentation as a fragment-flood attack rather than ordinary IP fragmentation.

A. Distributed NXDOMAIN flooding is a form of DDoS attack.

B. An IP fragment flood is a traffic-based attack.

C. DNS Flooding is a traffic-based attack.

D. TCP SYN Flooding is not a traffic-based attack.

Hint

Match the resource bottleneck or wireless mechanism to the relevant protection layer.

Answer D: TCP SYN Flooding is not a traffic-based attack.

The stated attacks exhaust resources through network traffic. Distributed NXDOMAIN flooding stresses resolver capacity using multiple sources; fragment floods abuse reassembly or packet-processing resources; DNS floods target query capacity. A SYN flood also uses network traffic, leaving incomplete TCP handshakes that consume connection state. Its mechanism differs from raw bandwidth exhaustion, but that does not make it non-traffic-based. D is therefore the false claim under the explicitly broad traffic-based meaning.

Why each choice is right or wrong

A - Incorrect: Multiple-source NXDOMAIN flooding qualifies as distributed denial of service.

B - Incorrect: Fragment flooding abuses processing or reassembly resources through traffic.

C - Incorrect: DNS flooding consumes query capacity through traffic.

D - Correct: SYN flooding is traffic-based, so this denial is false.

Primary note proof

Study-note section SEC06

> DoS exhausts or disables a resource; distributed DoS uses multiple sources, often bots or reflected/amplified responses. A SYN flood leaves many incomplete TCP handshakes occupying state. DNS flooding targets query capacity; NXDOMAIN attacks generate queries for nonexistent names to stress resolvers. Fragment floods abuse reassembly or packet-processing resources. Rate limits, SYN cookies, filtering and upstream capacity controls address different bottlenecks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 42-47

> • Denial of Service (DoS) Attack: Blocking access from
> legitimate users
> • Distributed DoS Attack: is accomplished by tricking
> routers into attacking a target or using Zombie hosts to

simultaneously attack a given target with large number of packets.

Question sources

Mock exam 4 · Question 36

Question 13 | Q00235

Which one of the following is true about Bell-LaPadula model security properties?

A. The * (star) Security Property states that a subject at a given security level may not read an object at a higher security level.

B. Users can create content only at or below their own security level (i.e. secret users can create secret or confidential files but may not create top secret files; no write-up).

C. The Strong Star Property states that subjects may write to objects with only a matching security level.

D. The Simple Security Property states that a subject at a given security level may not write to any object at a lower security level.

Hint

Apply the named model's read and write restrictions separately.

Answer C: Strong-star permits writes only at the matching level

SEC10 separates the properties: simple security is no read up, star is no write down and strong-star restricts writing to the same level. C correctly names the strong-star condition. A and D exchange the names of the simple and star properties, while B states no write up, contrary to Bell-LaPadula's confidentiality direction.

Why each choice is right or wrong

A - Incorrect: No read up is the simple-security rule, not the star property.

B - Incorrect: Bell-LaPadula prohibits writing down; this option instead asserts no write up and allows downward disclosure.

C - Correct: Strong-star permits writes only to objects at the subject's matching level.

D - Incorrect: No write down is the star rule, not the simple-security rule.

Primary note proof

Study-note section SEC10

Bell-LaPadula protects confidentiality. The simple-security rule prohibits reading above a subject's level; the star rule prohibits writing down. The strong-star variant permits writing only at the same level. With TS>S>C>U, an S subject can read C under the simple rule but cannot write C under star; compartments and discretionary restrictions may still deny access.

Study-note section SEC10

It is not a confidentiality model. “MAC” is an enforcement category and does not force every system to use Bell-LaPadula rather than another label policy.

Study-note section SEC10

Strict Biba protects integrity by preventing read down and write up: untrusted lower-integrity input should not contaminate higher-integrity results.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Computer Security module | PDF page 11 | lines 80-87

6. Access control
• Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Question sources

Mock exam 4 · Question 37

Question 14 | Q00236

Which statement most precisely identifies the programming defect underlying a buffer overflow, rather than a possible exploit outcome or a broad claim about operating-system families?

A. An overflow may allow attacker-supplied instructions to execute

B. The age or family name of an operating system alone establishes that it contains an exploitable overflow

C. All other statements precisely define the underlying defect

D. Buffer-overflow may remain as a bug in apps if boundary checks are not done fully

Hint

Identify the unsafe operation or input boundary rather than only a possible consequence.

Answer D: Buffer-overflow may remain as a bug in apps if boundary checks are not done fully

A buffer overflow writes beyond allocated bounds. Missing or incomplete boundary checks can leave that defect in application code, which is the role described by D. Executing attacker-supplied code is a possible consequence, not the underlying definition or a necessary outcome. An operating-system family name or age alone does not establish a specific vulnerable component. The revised question distinguishes cause from consequence without endorsing unsupported historical vendor claims.

Why each choice is right or wrong

A - Incorrect: This is a possible consequence rather than the underlying bounds-checking defect requested.

B - Incorrect: A family name or age is not evidence of a specific exploitable out-of-bounds write.

C - Incorrect: The other statements are not all precise descriptions of the underlying defect.

D - Correct: Incomplete bounds checking can leave an application buffer-overflow bug.

Primary note proof

Study-note section SEC25

> A buffer overflow writes beyond allocated bounds and can corrupt data or control information; code execution is a possible consequence, not its definition or an inevitable outcome. An array of eight bytes cannot store eight text characters plus a terminating zero. Bounds-check lengths and indexes, consider integer overflow, use safe abstractions and validate library return values. Canaries, non-executable memory and address randomization reduce exploitability but do not repair the underlying write.

Study-note section SEC25

> Security patches repair known weaknesses; a zero-day may lack a vendor fix, but segmentation and other mitigations can still help.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 53-59

• Example: If a company has antivirus software but does not use an anti-virus and keep the virus signatures up-to-date, this is vulnerability. The company is vulnerable to virus attacks. Similarly, if you fail to routinely update your operating systems or application software, these will remain vulnerable to software problems ("bugs") that have been identified and patched

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Mock exam 4 · Question 38

Question 15 | Q00237

Which one of the following is true about Firewalls?

A. Prevent a TCP Session Hijacking attack

B. Prevent a virus or malicious code from entering to our network

C. Can be configured to limit access based on port number or protocol

D. Detects and prevent an attack if attacker is within our network segment

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer C: Configure access limits by protocol and port

SEC23 explicitly gives protocol/port matching as a firewall function. It also describes bypass paths, insiders and infected removable media as limits, so firewall deployment does not guarantee all hijacking, malware or same-segment attacks are prevented. C is the configurable capability supported without those absolute guarantees.

Why each choice is right or wrong

A - Incorrect: A firewall does not universally prevent every session hijack, especially traffic outside its inspected or enforced paths.

B - Incorrect: Malware can arrive through bypass paths or permitted traffic; a firewall cannot guarantee excluding all malicious code.

C - Correct: Protocol and port fields can be matched in traffic-policy rules to restrict access.

D - Incorrect: An attacker sharing an internal segment may bypass the firewall's traffic path; detection/prevention is not guaranteed.

Primary note proof

Study-note section SEC23

A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Study-note section SEC23

Bypass paths, insiders, encrypted uninspected content and infected removable media limit firewall protection.

Study-note section SEC23

Stateful and next-generation devices can also filter ports; port filtering is not exclusive to one firewall category.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 22-34

3. Firewall
• a program or network devices (e.g. router) which filters access to a protected network from the internet connection.
• protects a local or network of systems from network-based security threats,
• monitors & controls incoming & outgoing network traffic based on predetermined security rules.
• Firewall Objectives
o Keep intruders, malicious code and unwanted traffic out
o Keep private and sensitive information in
o security wall b/n private (protected) NW & outside word.

Computer Security module | PDF page 11 | lines 35-56

• Categorizes of firewalls
a) Network firewalls: Filter traffic b/n two or more networks and run-on NW hardware.
✓ Implemented at a specific point in the network path and protects all computers on the internal side of the firewall from all computers on the external side of

the fire wall.
b) Host-based firewalls: Run on host computers and control network traffic in/out of those machines.
a) Packet Filtering Router: Examines TCP/IP header information of network data packets going in both directions. Applies set of rules to each incoming IP packets & then forwards or discards the packets.
b) Application-Level Gateway (application firewall / proxy server): runs special software that acts as a proxy for a service request. Proxy server receives requests for Web pages, accesses the Web server on behalf of the external client, and returns the requested pages to the users.
c)Circuit Level Gateway: Standalone Software that sets up two TCP connections (inside & outside). It determines which connections will be allowed.

Question sources

Mock exam 4 · Question 39

Question 16 | Q00238

Which one of the following is true about means of authentication?

A. Something the individual does: recognition by voice pattern, handwriting characteristics, and typing rhythm.

B. Something the individual possesses: fingerprint, retina, and face.

C. Something the individual is: electronic keycards, smart cards, and physical keys.

D. Something the individual knows: possess a token.

Hint

Separate identity claims from proof, and factor type from factor count.

Answer A: Behavioral characteristics: voice/handwriting/typing patterns

SEC07 distinguishes knowledge, possession and inherence, and recognizes behavioral biometrics under the textbook “does” category. A matches that classification, including handwriting and typing rhythm. The remaining options exchange possession and biometrics or call a possessed token knowledge. Voice-pattern recognition is biometric behavior/characteristic recognition, not possession of an external object.

Why each choice is right or wrong

A - Correct: The examples describe biometric behavior/patterns consistent with the textbook something-one-does category.

B - Incorrect: Fingerprints, retina and face are body characteristics, not possessed token/card objects.

C - Incorrect: Keycards, smart cards and physical keys are possession factors, not inherent body characteristics.

D - Incorrect: Possessing a token is a possession factor; knowledge means a secret such as a password or PIN.

Primary note proof

Study-note section SEC07

> Knowledge is a password/PIN, possession is a token/card, and inherence is a biometric such as a fingerprint. Behavioral biometrics include typing rhythm or handwriting; some textbooks call these “something one does.” A body characteristic is not “three-factor authentication”: one biometric is one factor. Password plus PIN remains one factor class; card plus PIN is two. An OTP’s factor depends on its delivery/generation mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 74-79

> • Most OS authentication is handled through user names and passwords. Biometric (e.g. voice, face, retina, iris, fingerprint) and physical token-based (swipe cards, pin-generating cards) authentication are sometimes used to augment simple passwords, but the costs and accuracy of the technology limit their adoption.

Question sources

Mock exam 4 · Question 40

Question 17 | Q00269

Which one of the following pairs is incorrect?

A. IP Spoofing - IPSec

B. TCP Session Hijacking - TLS

C. DNS Cache Poisoning - DNSSEC

D. Buffer overflow - HTTPS

Hint

Match the resource bottleneck or wireless mechanism to the relevant protection layer.

Answer D: HTTPS does not fix buffer overflows

SEC06 matches protections to the attacked layer: authenticated IPsec can resist IP impersonation, DNSSEC authenticates DNS data and TLS protects application transport against interception/tampering. HTTPS is HTTP over TLS; it cannot make an unsafe server memory copy bounds-safe. Thus D is the mismatched attack/defense pairing, without treating the other mechanisms as unconditional guarantees.

Why each choice is right or wrong

A - Incorrect: Authenticated IPsec can resist IP impersonation, so this is a relevant defense pairing.

B - Incorrect: TLS protects authenticated transport against interception/tampering relevant to session hijacking, under correct endpoint validation.

C - Incorrect: DNSSEC authenticates DNS data and is relevant to resisting false cached DNS answers.

D - Correct: HTTPS protects the channel but does not repair writes beyond a server buffer's allocated bounds.

Primary note proof

Study-note section SEC06

> Rogue access points imitate or bypass approved wireless service. Bluesnarfing steals Bluetooth information; bluejacking sends unsolicited Bluetooth messages. Phishing is not specific to wireless. Match a defense to its layer: authenticated IPsec can resist IP impersonation, DNSSEC authenticates DNS data, TLS authenticates/protects application transport, and bounds-safe code prevents overflows. HTTPS cannot fix an unsafe memory copy in a server.

Study-note section SEC25

> A buffer overflow writes beyond allocated bounds and can corrupt data or control information; code execution is a possible consequence, not its definition or an inevitable outcome.

Study-note section SEC21

> It protects HTTP paths, headers and content on the channel, but does not hide all metadata such as IP endpoints and traffic timing, nor protect data after a compromised endpoint decrypts it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 8 | lines 61-76

> Asymmetric key cryptography
> • also known as public-key encryption.
> • In public key cryptography,
> o unlike symmetric-key, there are two keys: a private key and a public key.
> ✓ The public key is announced to the public; whereas the private key is kept by the receiver.

o anyone can send messages using the public key of the receiver for encryption, but only the receiver uses his private key for decryption
✓ Secret transmission of key for decryption is not required
✓ Every entity can generate a key pair and release its public key
✓ Encryption and decryption keys are different but form a unique pair

Question sources

Mock exam 4 · Question 71

Question 18 | Q00341

Which statement is NOT a property of integrity under the primary CIA-goal classification?

A. Integrity can be compromised by Deception

B. Integrity can be compromised by Disclosure

C. Requires permitted modifications to be authorized

D. Protects resources against unauthorized alteration

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer B: Integrity can be compromised by Disclosure

Integrity concerns the correctness of information and protection against unauthorized or improper changes. False data introduced through deception, unauthorized alteration and violations of modification permissions can therefore threaten integrity. Pure disclosure means an unauthorized party learns information without changing it; that directly violates confidentiality. Availability instead concerns timely, usable access. Classifying the direct effect avoids treating every active attack as an integrity attack or assuming that one incident can affect only one security goal.

Why each choice is right or wrong

A - Incorrect: An integrity concern: deception can introduce false information and corrupt correctness.

B - Correct: Correct selection: pure disclosure directly compromises confidentiality rather than integrity.

C - Incorrect: An integrity concern: modification permissions distinguish authorized changes from unauthorized ones.

D - Incorrect: An integrity concern: unauthorized alteration directly damages information correctness.

Primary note proof

Study-note section SEC01

Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC03

The 4D classification uses disclosure, deception, disruption and usurpation, the last meaning unauthorized control.

Study-note section SEC03

Active attacks inject, alter, replay or disrupt.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 79-86

• Integrity: covers Data integrity and System integrity
o the prevention of unauthorized writing or modification of information.
o Integrity in a computer system means that there is an external consistency in the system - everything is as it is expected to be.
o Data integrity: the data stored on the computer is the same as what is intended.

Computer Security module | PDF page 4 | lines 50-61

• Types of Attacks on Data CIA (W. Stallings)
o Disclosure/Interception: unauthorized party snooping, or getting access to a resource, Attack on data confidentiality:
o Modification / Deception: unauthorized party modifying a resource, acceptance of false data, E.g., providing wrong data (attack on data integrity)
o Disruption/Interruption: destroyed/unavailable services/resources, DoS (attack on data availability)
o Fabrication: unauthorized party inserts a fake asset/resource, loss of confidentiality, authenticity and integrity of the message

Question sources

Computer Science model exam · Question 72

Computer Science model exam 2 · Question 88

University of Gondar model exam · Question 72

Question 19 | Q00342

Assume, you went to the Ethiopian airport to catch a flight and told the airport personnel your name. For sure, they would ask you some questions like your passport number and your reservation ticket number for some proof of your identity. Then what do we call this process?

A. Authentication

B. Authorization

C. Access control

D. All

Hint

Separate identity claims from proof, and factor type from factor count.

Answer A: Authentication

The passenger first claims a name, then is asked for evidence supporting that identity. Proving who one claims to be is authentication (PDF p.4), rather than deciding what the passenger may do.

Why each choice is right or wrong

A - Correct: Passport/ticket evidence tests the claimed identity. See PDF p.4.

B - Incorrect: Authorization would grant permissions after establishing identity. See PDF p.4.

C - Incorrect: Access control is the broader restriction mechanism, while the described step is identity proof. See PDF p.4.

D - Incorrect: All incorrectly treats permission decisions as already described. See PDF p.4.

Primary note proof

Study-note section SEC07

> A common access sequence is identify the subject, authenticate the claim, authorize the operation, then record relevant events. Identification is the username/claim, not its proof. When a sequence begins with authentication, distinguish it from the preceding identification stage. User-to-host means a user proves identity to a host; host-to-user means the host proves its identity to a user; the arrow names the prover and verifier.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Question sources

Computer Science model exam · Question 73

University of Gondar model exam · Question 73

Question 20 | Q00343

Assume, you went to the nearby commercial bank's automatic teller machine (ATM) to withdraw money. You inserted your ATM card into the machine, and the machine asked you to enter your personal identification number (PIN). Then, when you type your password, someone intercepts your ATM card number, and which of your security goals is compromised?

A. Integrity

B. Authorization

C. Confidentiality

D. Availability

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer C: Confidentiality

Intercepting the ATM number discloses information to an unauthorized party. The module maps interception to confidentiality (PDF pp.3-4); neither data alteration nor service denial is described.

Why each choice is right or wrong

A - Incorrect: No modification is stated. See PDF p.3, PDF p.4.

B - Incorrect: The event is disclosure rather than a permission-granting decision. See PDF p.3, PDF p.4.

C - Correct: Unauthorized learning of the card number compromises confidentiality. See PDF p.3, PDF p.4.

D - Incorrect: The user is not said to lose timely access to service. See PDF p.3, PDF p.4.

Primary note proof

Study-note section SEC01

Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC07

Password plus PIN remains one factor class; card plus PIN is two.

Study-note section SEC07

Knowledge is a password/PIN, possession is a token/card, and inherence is a biometric such as a fingerprint.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 71-78

• Confidentiality: Data confidentiality and Privacy
o is the prevention of unauthorized disclosure of information.
o In other words, confidentiality means keeping information private or safe.
o it may be important for military, business or personal reasons.
o it may also be known as privacy or secrecy.

Computer Security module | PDF page 4 | lines 50-61

• Types of Attacks on Data CIA (W. Stallings)
o Disclosure/Interception: unauthorized party snooping, or getting access to a resource, Attack on data confidentiality:
o Modification / Deception: unauthorized party modifying a resource, acceptance of false data, E.g., providing wrong data (attack on data integrity)
o Disruption/Interruption: destroyed/unavailable services/resources, DoS (attack on data availability)
o Fabrication: unauthorized party inserts a fake asset/resource, loss of confidentiality, authenticity and integrity of the message

Computer Security module | PDF page 3 | lines 87-91

• Availability:
o Assures that systems work promptly and service is not denied to authorized users.
o Information should be accessible and usable upon appropriate demand by an authorized user.

Question sources

Computer Science model exam · Question 74

University of Gondar model exam · Question 74

Question 21 | Q00344

Which attack can be used for monitoring children's activity?

A. Spyware

B. Ransomwares

C. Worms

D. Adware

Hint

Separate propagation, disguise and payload when classifying malicious software.

Answer A: Spyware-like monitoring capability

The capability described is monitoring user activity, which SEC04 associates with spyware. Ransomware locks or encrypts for extortion, worms replicate and adware displays unwanted advertisements. SEC05 adds an important context limit: monitoring with authorization is not automatically an attack. A matches the functionality, but the child-monitoring use alone does not establish malicious intent.

Why each choice is right or wrong

A - Correct: Spyware’s monitoring function matches collection of user activity, subject to the authorization caveat.

B - Incorrect: Ransomware denies access and extorts payment rather than primarily monitoring user activity.

C - Incorrect: A worm independently replicates; monitoring is not the defining capability.

D - Incorrect: Adware primarily displays unwanted advertisements, although some packages may also track activity.

Primary note proof

Study-note section SEC04

A virus replicates by attaching to a host program or document; execution of the infected host spreads it. A worm replicates independently, commonly across networks. A Trojan disguises malicious behavior as something useful and does not require self-replication. These labels describe different properties, so a malware package may contain several components.

Study-note section SEC04

Adware displays unwanted advertising and may also track users.

Study-note section SEC04

Ransomware denies access through encryption/locking and extortion.

Related reading (context, not primary note proof)

Study-note section SEC05

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 21-35

• Malware Attack: A generic term for software that has malicious purpose.
o Viruses
o Worms
o Trojan horses
o Spy-wares
o Zombie
o Botnets
o Bacterium
o Logic bomb
o Backdoor(Trapdoor)
o Adware
o Flooders (DoS client)
o Key loggers
o Rootkit

Computer Security module | PDF page 5 | lines 65-74

• Reconnaissance: The hackers first identifies a target to launch an attack, extract maximum information regarding this target, understand its vulnerabilities, and then only explore the best ways to exploit it.
o Examples: Credential Theft using Keyloggers and Spyware; Identity Theft using spam e-mail, phishing attack and social engineering attack
• Access: Ones the attacker Reconnaissance the target, S/he starts a successful exploitation and continued access to the system. Then, the attacker is in control of the target.

Question sources

Computer Science model exam · Question 75

University of Gondar model exam · Question 75

Question 22 | Q00345

_____is a type of data link layer attack in TCP/IP protocol stack

A. ARP spoofing

B. DNS spoofing

C. SYNC-FLOOD attack

D. All except c

Hint

Identify the trust claim or communication mapping being abused.

Answer A: ARP spoofing

ARP spoofing targets the local mapping between IPv4 and link-layer MAC addresses, making A the data-link-oriented attack among these choices. DNS spoofing targets name resolution rather than local link associations, and SYN flooding abuses TCP handshake state. D incorrectly includes DNS spoofing in the data-link category.

Why each choice is right or wrong

A - Correct: ARP spoofing poisons the local IPv4-to-MAC mapping at the network/link addressing boundary.

B - Incorrect: DNS spoofing falsifies name-resolution data rather than manipulating the local data-link association.

C - Incorrect: The likely SYN-flood reference concerns TCP handshake state at transport scope, not link addressing.

D - Incorrect: This includes DNS spoofing, which is not a data-link mapping attack.

Primary note proof

Study-note section SEC05

> Masquerade claims another identity. ARP spoofing poisons local IPv4-to-MAC associations; DNS spoofing/cache poisoning supplies false name resolutions; IP spoofing falsifies a source address. An intermediary attack intercepts and can alter exchanges. A replay reuses a valid old message; nonces, timestamps and sequence checks can reject it. Session hijacking steals or predicts a valid session token. Protect transport, regenerate session IDs after login, restrict cookies and check authorization server-side on every sensitive request.

Study-note section SEC06

> A SYN flood leaves many incomplete TCP handshakes occupying state.

Study-note section SEC06

DNS flooding targets query capacity; NXDOMAIN attacks generate queries for nonexistent names to stress resolvers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 42-47

• Denial of Service (DoS) Attack: Blocking access from legitimate users
• Distributed DoS Attack: is accomplished by tricking routers into attacking a target or using Zombie hosts to simultaneously attack a given target with large number of packets.

Question sources

Computer Science model exam · Question 76

University of Gondar model exam · Question 76

Question 23 | Q00346

Which statement correctly distinguishes the host-program dependence of viruses and worms? Here 'host' means a carrier program or document, not a computer.

A. Worms requires a host to spread while viruses doesn't require

B. Worms requires a user action for replicate while viruses doesn't require

C. Virus requires a host to spread while Worms doesn't require

D. Worms spreading speed is slower than Virus Spreading speed

Hint

Separate propagation, disguise and payload when classifying malicious software.

Answer C: Virus requires a host to spread while Worms doesn't require

A virus replicates by attaching to a carrier program or document and spreading when that infected host executes. A worm replicates independently of such a carrier. This makes C the correct direction and makes A a reversal. B wrongly makes user action a defining requirement for worm replication while denying the virus's dependence on infected-host execution. D proposes a general speed ranking, which is not a defining consequence of either replication mechanism.

Why each choice is right or wrong

A - Incorrect: Incorrect: this reverses the definitions; viruses need carrier programs/documents and worms replicate independently.

B - Incorrect: Incorrect: independent worm replication does not require a user to launch each copy; virus propagation depends on infected-host execution.

C - Correct: Correct: viruses attach to carrier hosts, whereas worms replicate independently of such carriers.

D - Incorrect: Incorrect: propagation speed does not follow from the carrier-versus-independent distinction and cannot define the requested host dependence.

Primary note proof

Study-note section SEC04

A virus replicates by attaching to a host program or document; execution of the infected host spreads it. A worm replicates independently, commonly across networks. A Trojan disguises malicious behavior as something useful and does not require self-replication. These labels describe different properties, so a malware package may contain several components.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 21-35

• Malware Attack: A generic term for software that has
malicious purpose.
o Viruses
o Worms
o Trojan horses
o Spy-wares
o Zombie
o Botnets
o Bacterium
o Logic bomb
o Backdoor(Trapdoor)
o Adware
o Flooders (DoS client)
o Key loggers
o Rootkit

Question sources

Computer Science model exam · Question 77

University of Gondar model exam · Question 77

Question 24 | Q00358

Which general security-control category enforces allow/block rules for incoming and outgoing network traffic? Choose the functional security category rather than a general forwarding or relay role.

A. Proxy

B. Firewall

C. Gateway

D. Router

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer B: Firewall

A firewall is the general control that enforces a network traffic policy, allowing or blocking traffic according to rules. A router's basic role is forwarding IP packets; a proxy relays application requests, and gateway is a broader connection or mediation term. These devices can include firewall capabilities, so their hardware or networking names alone do not make mutually exclusive implementations. The stem asks for the functional security category, which makes firewall the precise answer.

Why each choice is right or wrong

A - Incorrect: A proxy's general role is relaying application exchanges; it may also implement firewall policy but is not the general category requested.

B - Correct: Correct: firewall names the general traffic-policy enforcement control.

C - Incorrect: Gateway is a broad networking or mediation role and does not by itself specify security-rule enforcement.

D - Incorrect: A router forwards IP packets; filtering added to a router is a firewall function rather than an implication of routing alone.

Primary note proof

Study-note section SEC23

> A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Study-note section SEC-S03

> A router forwards IP packets between networks; forwarding alone is different from detecting suspicious behavior or enforcing a firewall's security policy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 22-34

3. Firewall
• a program or network devices (e.g. router) which filters access to a protected network from the internet connection.
• protects a local or network of systems from network-based security threats,
• monitors & controls incoming & outgoing network traffic based on predetermined security rules.
• Firewall Objectives
o Keep intruders, malicious code and unwanted traffic out
o Keep private and sensitive information in
o security wall b/n private (protected) NW & outside word.

Computer Security module | PDF page 11 | lines 35-56

• Categorizes of firewalls
a) Network firewalls: Filter traffic b/n two or more networks and run-on NW hardware.
✓ Implemented at a specific point in the network path and protects all computers on the internal side of the firewall from all computers on the external side of the fire wall.
b) Host-based firewalls: Run on host computers and control network traffic in/out of those machines.
a) Packet Filtering Router: Examines TCP/IP header information of network data packets going in both directions. Applies set of rules to each incoming IP packets & then forwards or discards the packets.
b) Application-Level Gateway (application firewall / proxy server): runs special software that acts as a proxy for a service request. Proxy server receives requests for Web pages, accesses the Web server on behalf of the external client, and returns the requested pages to the users.
c)Circuit Level Gateway: Standalone Software that sets up two TCP connections (inside & outside). It determines which connections will be allowed.

Question sources

Computer Science model exam · Question 89

University of Gondar model exam · Question 88

Question 25 | Q00642

Firewall in computer network is

A. physical boundary of network

B. an operating system of network

C. A system designed to prevent unauthorized access

D. A web browsing software

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer C: A system designed to prevent unauthorized access

A firewall enforces rules controlling network traffic to prevent unauthorized access. It may be software or a network device and therefore need not be a physical boundary. An operating system is not the name of this specific traffic-policy function, and a web browser requests or displays web content rather than serving as the general network-access enforcement control. C captures the firewall's security purpose, subject to the normal limits of configured rules and available inspection.

Why each choice is right or wrong

A - Incorrect: Incorrect: traffic-policy enforcement is a logical control and need not be a physical boundary.

B - Incorrect: Incorrect: firewall names the traffic-policy enforcement function, not an entire network operating system.

C - Correct: Correct: preventing unauthorized network access through traffic rules is the firewall purpose.

D - Incorrect: Incorrect: browsing web content is different from enforcing network traffic policy.

Primary note proof

Study-note section SEC23

> A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 22-34

> 3. Firewall
> • a program or network devices (e.g. router) which filters

access to a protected network from the internet connection.
• protects a local or network of systems from network-based security threats,
• monitors & controls incoming & outgoing network traffic based on predetermined security rules.
• Firewall Objectives
o Keep intruders, malicious code and unwanted traffic out
o Keep private and sensitive information in
o security wall b/n private (protected) NW & outside word.

Question sources

Computer Science model exam · 2015 · Question 111

Question 26 | Q00914

Assume that we are able to specify that Mr. Getahun can view the database record, but cannot change the value of the record. Which security management technique was used in this scenario?

A. Access Control

B. Availability

C. ¢.
Non-repudiation

D. Integrity

Hint

Identify subject, object and permitted operation before comparing policy models.

Answer A: Access Control

Permitting read while denying write is a subject/object/operation permission assignment, exactly the access-enforcement model (PDF p.11). Integrity is a protected goal, not the mechanism requested.

Why each choice is right or wrong

A - Correct: Access control authorizes particular operations on an object. See PDF p.3, PDF p.11.

B - Incorrect: Availability means timely usable service and access for authorized users (PDF page3), rather than selecting the allowed operation on a particular record.

C - Incorrect: Non-repudiation concerns proving an event or action so sender or receiver cannot deny it (PDF page4), rather than assigning read/write permissions.

D - Incorrect: Integrity is the desired protection against unauthorized changes, not the permission-setting mechanism. See PDF p.3, PDF p.11.

Primary note proof

Study-note section SEC09

Subjects request operations on objects. An ACL associates an object/resource with allowed subjects or actions; a network ACL matches traffic such as address, protocol and port. Classification and filtering are direct effects; high availability or malware detection is not guaranteed by an ACL.

Study-note section SEC01

Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it.

Study-note section SEC09

Allowing SELECT but denying UPDATE is an access-control decision that helps integrity; it does not by itself enforce all valid business-state transitions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 80-87

6. Access control
• Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Computer Security module | PDF page 3 | lines 79-86

• Integrity: covers Data integrity and System integrity
o the prevention of unauthorized writing or modification of information.
o Integrity in a computer system means that there is an external consistency in the system - everything is as it is expected to be.
o Data integrity: the data stored on the computer is the same as what is intended.

Computer Security module | PDF page 4 | lines 5-13

Additional objectives:
• Non-repudiation: is the prevention of either the sender or the receiver denying a transmitted message.
o It is the ability to prove the occurrence of a claimed event or action and its originating entities
o It assures that a sender of data is provided with proof of delivery and the recipient is provided with proof of the

sender's identity, so neither can later deny having processed the data.

Computer Security module | PDF page 3 | lines 87-91

• Availability:
o Assures that systems work promptly and service is not denied to authorized users.
o Information should be accessible and usable upon appropriate demand by an authorized user.

Question sources

Computer Science Exit Exam · 2018 · Question 27

Question 27 | Q00920

Which statement is NOT true about digital signatures?

A. A digital signature is not, by itself, a confidentiality-encryption mechanism

B. The sender signs a message or its encoded digest using its private signing key

C. The sender generates a digital signature using its public verification key

D. Digital signature provides authentication services

Hint

Separate a digest, shared-secret authentication and publicly verifiable signing.

Answer C: The sender generates a digital signature using its public verification key

A digital signature is generated with the signer's private signing key and checked with the corresponding public verification key. Signing may operate on a message or an encoded digest, depending on the algorithm. It is not generically private-key encryption, and it does not by itself hide the message. Under an authentic key binding and secure private-key custody, verification supports origin authentication and integrity. The public verification key cannot generate the signer's signature, making C false.

Why each choice is right or wrong

A - Incorrect: True: a signature supports origin and integrity checks without itself concealing the content.

B - Incorrect: True: signature generation uses the private signing key on the message or its encoded digest.

C - Correct: Correct selection: the public verification key verifies signatures; it does not generate them.

D - Incorrect: True: signature verification supports origin authentication under the stated key-binding assumptions.

Primary note proof

Study-note section SEC20

A signature algorithm signs a message or its encoded digest with a private key and verifies with the public key. It is not generically “encrypting with a private key,” especially for DSA/ECDSA. Verification checks origin/integrity under key-identity and key-custody assumptions; it does not conceal content. Password verification uses a salt and deliberately expensive password hash. Historical MD5/SHA-1 examples do not justify their use for new collision-resistant signatures.

Study-note section SEC18

For a signature, A signs with A’s private signing key and others verify with A’s public key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 7-20

(4) Digital Signature Algorithm (DSA):
o PKI relies on DS, which uses public key cryptography.
o The basic idea is that private key is only known by
that entity & used for signing.
✓ Public key derived from it: used for verifying
signatures but cannot be used to sign.
✓ It is available to anyone, and is typically included in
the certificate document.
o Digital Signature procedure
• Sender encrypts message with its private key
• Receiver can decrypt using sender’s public key
• This authenticates sender, who has the matching key
• Does not give privacy of data i.e. Decrypt key is
public key.

Question sources

Computer Science Exit Exam · 2018 · Question 33

Question 28 | Q00923

Which one of the following party performs the technical evaluation work, using the evidence supplied by the developers, and additional testing of the product, to confirm that it satisfies the functional and assurance requirements specified in the security target?

A. Evaluator

B. Certifier

C. Sponsor

D. Developer

Hint

Distinguish requirements, technical evaluation, evidence and certification decisions.

Answer A: Evaluator

SEC29 assigns technical examination of developer evidence, testing and vulnerability analysis against Security Target claims to evaluators. The certification body oversees/reviews evaluation and issues the certification decision; sponsors commission the work and developers supply the product/evidence. A therefore names the party performing the actual technical evaluation rather than those adjacent roles.

Why each choice is right or wrong

A - Correct: Evaluators review developer evidence and perform tests/analysis against the stated functional and assurance claims.

B - Incorrect: The certifier oversees/reviews evaluation and issues the certification decision rather than being the named technical evaluator.

C - Incorrect: The sponsor commissions the work; that role does not itself perform the technical evaluation.

D - Incorrect: The developer supplies the product and supporting evidence for independent examination.

Primary note proof

Study-note section SEC29

> Under Common Criteria, define the Target of Evaluation (TOE), its configuration and environment. A Security Target (ST) states threats, assumptions, objectives, functional requirements and assurance requirements; a Protection Profile describes reusable requirements for a product category. Evaluators examine developer evidence, test and analyze vulnerabilities against those claims. A certification body oversees/reviews the evaluation and issues the certification decision; sponsors commission work and developers supply product/evidence.

Study-note section SEC29

> A certificate applies to a specified product/configuration and claims, not immunity to all future attacks.

Study-note section SEC29

> An Evaluation Assurance Level packages assurance requirements; EAL1-7 express increasing evaluation rigor, not a universal ranking of security functions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

> ✓ Vulnerability Assessment: the process of scanning
> (locating & reporting) vulnerabilities to provide a
> way to detect and resolve security problems by

ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Computer Security module | PDF page 10 | lines 85-90

o Penetration Test (pen test):
✓ To identify both weaknesses (vulnerabilities) & strengths, to enable a full risk assessment.
✓ Determine whether a system is vulnerable, defenses were sufficient and which defenses (if any) the test defeated.

Question sources

Computer Science Exit Exam · 2018 · Question 36

Question 29 | Q00941

Compare two incidents at Registrar X. In incident I, Negesa composes a fresh request and presents it as coming from Dereje. In incident II, Negesa retransmits, without changing it, a valid request that Dereje genuinely sent earlier. Which statement correctly distinguishes the observed attack mechanisms?

A. I makes the registrar unavailable, whereas II leaves the service available.

B. I alters the content of Dereje's existing request, whereas II leaves that request unchanged.

C. I creates a fresh message under Dereje's identity, whereas II repeats Dereje's authentic old message.

D. I repeats an authentic old message, whereas II creates a fresh message under a false identity.

Hint

Compare the claimed origin, message freshness and whether an existing message was changed.

Answer C: I creates a fresh message under Dereje's identity, whereas II repeats Dereje's authentic old message.

Incident I introduces a newly composed message under another person's identity, which is masquerade. Incident II reuses a valid old message, which is replay. The relevant distinction is false origin for a fresh message versus reuse of an authentic earlier one. Neither incident describes service denial, and I does not modify an existing request. D reverses the two histories. Replay can also support impersonation in a larger attack, so this comparison classifies the specific observed mechanisms.

Why each choice is right or wrong

A - Incorrect: Neither history includes service exhaustion or interruption; service availability does not distinguish these incidents.

B - Incorrect: I creates a new request rather than altering an existing one; message modification is a different mechanism.

C - Correct: I's fresh false-origin request demonstrates masquerade, while II's unchanged reuse of an authentic earlier request demonstrates replay.

D - Incorrect: This reverses the observed histories: II reuses the old request, while I creates the fresh false-origin request.

Primary note proof

Study-note section SEC05

Masquerade claims another identity.

Study-note section SEC05

A replay reuses a valid old message; nonces, timestamps and sequence checks can reject it.

Study-note section SEC03

Interception copies or observes data, modification alters it, interruption blocks it, and fabrication inserts false material.

Study-note section SEC06

DoS exhausts or disables a resource; distributed DoS uses multiple sources, often bots or reflected/amplified responses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 50-61

• Types of Attacks on Data CIA (W. Stallings)
o Disclosure/Interception: unauthorized party snooping, or getting access to a resource, Attack on data confidentiality:
o Modification / Deception: unauthorized party modifying a resource, acceptance of false data, E.g., providing wrong data (attack on data integrity)
o Disruption/Interruption: destroyed/unavailable services/resources, DoS (attack on data availability)
o Fabrication: unauthorized party inserts a fake asset/resource, loss of confidentiality, authenticity and integrity of the message

Computer Security module | PDF page 5 | lines 65-74

• Reconnaissance: The hackers first identifies a target to launch an attack, extract maximum information regarding this target, understand its vulnerabilities, and then only explore the best ways to exploit it.
o Examples: Credential Theft using Keyloggers and

Spyware; Identity Theft using spam e-mail, phishing attack and social engineering attack
• Access: Ones the attacker Reconnaissance the target, S/he starts a successful exploitation and continued access to the system. Then, the attacker is in control of the target.

Question sources

Computer Science Exit Exam · 2018 · Question 54

Question 30 | Q00963

Which general security-control category enforces allow/block rules on incoming and outgoing network traffic? Choose the general enforcement category rather than a relay, tunnel or observing role.

A. Intrusion detection system (IDS)

B. Proxy server

C. Virtual Private Network (VPN)

D. Firewall

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer D: Firewall

A firewall is the general traffic-policy enforcement category that permits or blocks traffic under rules. An IDS observes and alerts, whereas a proxy relays application requests and may additionally implement policy. A VPN establishes a logical private connection whose security depends on its tunnel; that does not itself define allow/block filtering. Because implementations can combine these functions, the revised stem asks for the general enforcement category rather than treating all device types as mutually exclusive.

Why each choice is right or wrong

A - Incorrect: Incorrect: an IDS observes and alerts; the requested role enforces traffic permission decisions.

B - Incorrect: Incorrect: proxy names an application-relay role, even when a particular proxy also implements firewall policy.

C - Incorrect: Incorrect: a VPN provides a logical private connection; that role alone does not specify allow/block traffic enforcement.

D - Correct: Correct: firewall names the general traffic-policy enforcement category.

Primary note proof

Study-note section SEC23

A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Study-note section SEC24

An IDS observes and alerts; an IPS can enforce blocking, usually inline.

Study-note section SEC22

A VPN is a logical private connection; its actual security depends on the chosen authenticated tunnel.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 22-34

3. Firewall
• a program or network devices (e.g. router) which filters access to a protected network from the internet connection.
• protects a local or network of systems from network-based security threats,
• monitors & controls incoming & outgoing network traffic based on predetermined security rules.
• Firewall Objectives
o Keep intruders, malicious code and unwanted traffic out
o Keep private and sensitive information in
o security wall b/n private (protected) NW & outside word.

Computer Security module | PDF page 11 | lines 35-56

• Categorizes of firewalls
a) Network firewalls: Filter traffic b/n two or more networks and run-on NW hardware.
✓ Implemented at a specific point in the network path and protects all computers on the internal side of the firewall from all computers on the external side of the fire wall.
b) Host-based firewalls: Run on host computers and control network traffic in/out of those machines.
a) Packet Filtering Router: Examines TCP/IP header information of network data packets going in both directions. Applies set of rules to each incoming IP packets & then forwards or discards the packets.
b) Application-Level Gateway (application firewall /

proxy server): runs special software that acts as a proxy for a service request. Proxy server receives requests for Web pages, accesses the Web server on behalf of the external client, and returns the requested pages to the users.
c)Circuit Level Gateway: Standalone Software that sets up two TCP connections (inside & outside). It determines which connections will be allowed.

Computer Security module | PDF page 4 | lines 89-99

o Encryption: primary control that protects CIA, by which Cleartext scambled into ciphertext.
o Software controls: password checker, virus scanner, IDS (intrusion detection system)
o Hardware controls: provide higher degree of security; Locks, Smart cards, dongles, hadware keys...
o Policies and procedures: Alignment with users' legal and ethical standards; eg. password policy
o Physical controls: natural and man-made disaster protection; walls, locks, guards, security, cameras, backup copies and archives

Question sources

Computer Science Exit Exam · 2018 · Question 76

Question 31 | Q00972

Which one of the following is WRONGLY stated about an importance of security policy?

A. To help minimize risk.

B. To ensure the confidentiality, integrity and availability of data.

C. .
To coordinate and enforce a security program across an organization.

D. To use the system in a passive way

Hint

Distinguish organizational rules, operational steps and recovery goals.

Answer D: To use the system in a passive way

A security policy regulates resource protection (PDF p.3); assurance reduces risk and preserves protection goals (p.2). Merely using a system passively neither establishes protection rules nor coordinates enforcement.

Why each choice is right or wrong

A - Incorrect: Minimizing risk is consistent with regulating protection. See PDF p.2, PDF p.3.

B - Incorrect: Protecting CIA is consistent with security and assurance goals. See PDF p.2, PDF p.3.

C - Incorrect: Coordinating and enforcing organizational protection is consistent with a common rule set. See PDF p.2, PDF p.3.

D - Correct: Passive system use is not a security-policy purpose derived from the definition. See PDF p.2, PDF p.3.

Primary note proof

Study-note section SEC33

> A security policy states responsibilities and rules; a procedure supplies the steps. Policies coordinate protection, reduce risk and define acceptable use, access review, incident handling and recovery. They do not prescribe merely “passive system use.” Awareness training teaches reporting and independent verification; enforcement and usable procedures make it operational.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 41-43

> • Security Policy: A set of rules and practices that specify or regulate how a system or organization protects sensitive and critical system resources.

Computer Security module | PDF page 2 | lines 77-83

> • IA: measures that protect and defend information and ISs by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation.
> • Information protection or InfoSec: protection of information and ISs from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction.

Question sources

Computer Science Exit Exam · 2018 · Question 85

Question 32 | Q00975

Which one of the following computer security goal prevents the disclosure of sensitive information to unauthorized users or systems on computer networks?

A. Confidentiality

B. Integrity

C. Cyberspace

D. Availability

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer A: Confidentiality

The stem restates prevention of unauthorized disclosure, the confidentiality definition on PDF p.3. It asks about secrecy rather than correctness or access timing.

Why each choice is right or wrong

A - Correct: Confidentiality is explicitly prevention of unauthorized disclosure. See PDF p.3.

B - Incorrect: Integrity prevents unauthorized writing/modification. See PDF p.3.

C - Incorrect: Cyberspace is not one of the module's CIA goals. See PDF p.3.

D - Incorrect: Availability assures usable access for authorized users. See PDF p.3.

Primary note proof

Study-note section SEC01

> Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC01

> Data security protects information at rest, in use and in transit, including accidental corruption as well as intentional attacks.

Study-note section SEC01

> Information assurance includes confidence, risk management, protection, detection, response and restoration across the information lifecycle.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 71-78

• Confidentiality: Data confidentiality and Privacy
o is the prevention of unauthorized disclosure of information.
o In other words, confidentiality means keeping information private or safe.
o it may be important for military, business or personal reasons.
o it may also be known as privacy or secrecy.

Computer Security module | PDF page 3 | lines 79-86

• Integrity: covers Data integrity and System integrity
o the prevention of unauthorized writing or modification of information.
o Integrity in a computer system means that there is an external consistency in the system - everything is as it is expected to be.
o Data integrity: the data stored on the computer is the same as what is intended.

Computer Security module | PDF page 3 | lines 87-91

• Availability:
o Assures that systems work promptly and service is not denied to authorized users.
o Information should be accessible and usable upon appropriate demand by an authorized user.

Question sources

Computer Science Exit Exam · 2018 · Question 88

Question 33 | Q01175

Which of the following primarily provides detective evidence that can reveal an attack, rather than preventing access or disclosure?

A. Access Control

B. Encryption

C. Audit log

D. Password authentication

Hint

Distinguish weaknesses, potential harmful events, actual attempts and the effect of a control.

Answer C: Audit log

An audit log records events that can reveal suspicious activity during review or analysis and can support accountability. Its primary role in this comparison is detective evidence. Access control restricts permitted actions, password authentication checks an identity claim, and encryption limits disclosure to unauthorized readers. These are principally preventive roles in the question's comparison. A log does not automatically block an attack, and its usefulness depends on reliable recording, protection and analysis.

Why each choice is right or wrong

A - Incorrect: Access control primarily prevents actions that exceed a subject's permissions.

B - Incorrect: Encryption primarily prevents unauthorized disclosure rather than revealing attack activity through event records.

C - Correct: Correct: an audit log provides event evidence that can reveal attacks and support accountability.

D - Incorrect: Password authentication primarily checks identity before access; it is not the offered detective event record.

Primary note proof

Study-note section SEC02

> Preventive controls block events, detective controls reveal them, corrective/recovery controls restore safe operation, and deterrents discourage attempts. Encryption is preventive for disclosure; an audit log is principally detective/accountability evidence. Physical, technical and administrative are control categories, not competing stages. An external firewall cannot replace a written policy or a locked equipment room.

Study-note section SEC07

> Knowledge is a password/PIN, possession is a token/card, and inherence is a biometric such as a fingerprint.

Study-note section SEC07

> A common access sequence is identify the subject, authenticate the claim, authorize the operation, then record relevant events.

Study-note section SEC09

> Least privilege gives only needed rights, for needed time.

Related reading (context, not primary note proof)

Study-note section SEC12

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 28-35

• Accountability
o Ensuring an entity's action is traceable uniquely to that entity.
o Ascertains the responsibility of an entity (like a person) for its actions and decisions.
o All relevant activities events and operations on a system, e.g., failed and successful authentication attempts, are recorded in a log.

Computer Security module | PDF page 4 | lines 82-84

o Detect attack: accept that an attack will occur; the goal is to determine that an attack is under way, or has occurred, and report it.

Computer Security module | PDF page 4 | lines 89-99

o Encryption: primary control that protects CIA, by which Cleartext scambled into ciphertext.
o Software controls: password checker, virus scanner, IDS (intrusion detection system)
o Hardware controls: provide higher degree of security; Locks, Smart cards, dongles, hadware keys...
o Policies and procedures: Alignment with users' legal and ethical standards; eg. password policy
o Physical controls: natural and man-made disaster protection; walls, locks, guards, security, cameras, backup copies and archives

Question sources

Computer Science Exit Exam practice questions · Question 36

Question 34 | Q01189

The process of recovering the original message without the detailed knowledge of the secret information is called:

A. Decryption

B. Decipher

C. Cryptanalysis

D. A and B

Hint

Trace message transformations and separate data, keys, algorithms and codebreaking.

Answer C: Cryptanalysis

PDF p.6 distinguishes decryption (recovering plaintext) from cryptanalysis (breaking ciphertext without the key). The explicit absence of secret knowledge selects cryptanalysis.

Why each choice is right or wrong

A - Incorrect: Decryption is ordinary plaintext recovery and omits the without-key distinction. See PDF p.6.

B - Incorrect: Decipher is the same ordinary recovery operation. See PDF p.6.

C - Correct: Cryptanalysis explicitly concerns breaking ciphertext without knowing the key. See PDF p.6.

D - Incorrect: Combining two ordinary recovery synonyms does not supply the required without-key meaning. See PDF p.6.

Primary note proof

Study-note section SEC12

> Plaintext is the original message, ciphertext its encrypted representation, a key is input to an algorithm, encryption computes C=E_k(P), and decryption recovers P=D_k(C). The five conventional symmetric ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. Cryptanalysis seeks recovery or other breaks without the authorized secret; cryptology includes cryptography and cryptanalysis. Steganography hides that a message exists; encryption hides its meaning.

Study-note section SEC12

> Security depends on algorithm, key size, randomness, mode and implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 2-12

> o Encipher (encrypt) - the process of converting
> plaintext to ciphertext
> o Decipher (decrypt) - recovering plaintext from
> ciphertext
> o Cryptography - the study of encryption
> principles/methods
> o Cryptanalysis (codebreaking) - the study of
> principles/ methods of breaking ciphertext without
> knowing key
> o Cryptology - the field of study which deals with both
> cryptography and cryptanalysis

Question sources

Computer Science model exam 2 · Question 10

Question 35 | Q01203

Which of the following statements is true about crackers?

A. They attempt to break the system without permissions

B. They are legally employed to break into the system

C. They reveal the information after breaking into the system

D. All

Hint

Classify the action actually described and distinguish observation, alteration, disruption and identity claims.

Answer A: They attempt to break the system without permissions

Cracking is malicious hacking; intrusion is unauthorized access (PDF p.5). By contrast authorized simulated attacks are penetration testing (p.10). A matches unauthorized breaking, and neither legal employment nor later disclosure defines a cracker.

Why each choice is right or wrong

A - Correct: Attempted unauthorized system breaking matches malicious intrusion. See PDF p.5, PDF p.10.

B - Incorrect: Legally authorized testing belongs to penetration testing, not the definition of cracking. See PDF p.5, PDF p.10.

C - Incorrect: Revealing information afterward is not a necessary defining property of malicious intrusion. See PDF p.5, PDF p.10.

D - Incorrect: B and C are not defining truths, so All cannot hold. See PDF p.5, PDF p.10.

Primary note proof

Study-note section SEC03

> White-hat testing requires authorization and scope; black-hat activity is malicious or unauthorized. A competitor may seek commercial secrets; hacktivists pursue a cause. “Hacker” alone does not establish legality. Insider versus outsider describes origin/trust position, not the specific mechanism.

Study-note section SEC27

> Vulnerability assessment finds and prioritizes weaknesses; authorized penetration testing attempts to demonstrate consequences.

Study-note section SEC27

> Host/network/database scanners see different exposures and have false results.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 37-43

• Hacking /Intrusion/ Attack: any attempt to intrude or gain unauthorized access to your system either via some operating system flaw or other means. The purpose may or may not be for malicious purposes.
• Cracking: is hacking conducted for malicious purposes.
• Denial of Service (DoS) Attack: Blocking access from legitimate users

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Computer Science model exam 2 · Question 24

Question 36 | Q01207

Which basic firewall category makes stateless decisions from packet-header fields such as protocol and port numbers, without tracking connection state or terminating an application exchange?

A. Packet Filtering

B. Proxy Service

C. Stateful Inspection

D. Threat-Focused NGFW

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer A: Packet Filtering

Stateless packet filtering matches packet headers such as addresses, protocol and ports. Stateful inspection adds connection context, while an application proxy terminates and relays application exchanges. Next-generation products can also filter ports, but that capability alone does not make them the basic stateless header-only category requested. The additional constraints distinguish the model rather than incorrectly claiming port filtering is exclusive to one product type.

Why each choice is right or wrong

A - Correct: This is the stateless header-matching category.

B - Incorrect: A proxy terminates and relays application exchanges.

C - Incorrect: Stateful inspection tracks connection context, which the stem excludes.

D - Incorrect: A next-generation firewall is not the basic header-only stateless category requested.

Primary note proof

Study-note section SEC23

A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Study-note section SEC23

Stateful and next-generation devices can also filter ports; port filtering is not exclusive to one firewall category.

Study-note section SEC23

Hardware firewall does not imply any fixed number of ports.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 35-56

• Categorizes of firewalls
a) Network firewalls: Filter traffic b/n two or more networks and run-on NW hardware.
✓ Implemented at a specific point in the network path and protects all computers on the internal side of the firewall from all computers on the external side of the fire wall.
b) Host-based firewalls: Run on host computers and control network traffic in/out of those machines.
a) Packet Filtering Router: Examines TCP/IP header information of network data packets going in both directions. Applies set of rules to each incoming IP packets & then forwards or discards the packets.
b) Application-Level Gateway (application firewall / proxy server): runs special software that acts as a proxy for a service request. Proxy server receives requests for Web pages, accesses the Web server on behalf of the external client, and returns the requested pages to the users.
c)Circuit Level Gateway: Standalone Software that sets up two TCP connections (inside & outside). It determines which connections will be allowed.

Question sources

Computer Science model exam 2 · Question 28

Question 37 | Q01213

What will be the encrypted message if the message "ATTACK" is encrypted by Caesar's cipher with key = 25?

A. ZSSZBJ

B. ZGGZXP

C. ZTTZAK

D. BGGBDL

Hint

Separate symbol replacement from position changes and apply the alphabet and modular direction.

Answer A: ZSSZBJ

Using the additive shift on Z26 (PDF p.6), take A=0 and encryption C=(P+25) mod 26. ATTACK gives (0,19,19,0,2,10) -> (25,18,18,25,1,9) -> ZSSZBJ. This is a one-position backward shift because 25=-1 mod 26.

Why each choice is right or wrong

A - Correct: ZSSZBJ agrees at all six positions. See PDF p.6.

B - Incorrect: ZGGZXP has correct initial A->Z but T->G instead of S. See PDF p.6.

C - Incorrect: ZTTZAK leaves T unchanged and maps C incorrectly. See PDF p.6.

D - Incorrect: BGGBDL begins A->B, a forward-one shift rather than +25. See PDF p.6.

Primary note proof

Study-note section SEC13

> Substitution replaces symbols; transposition changes their positions. A monoalphabetic substitution preserves one mapping throughout the message and leaks frequency patterns despite a large nominal key space. Classical ciphers are teaching devices, not modern data protection.

Study-note section SEC13

> With A=0 through Z=25, Caesar encryption is C=(P+k) mod26 and decryption P=(C-k) mod26.

Study-note section SEC13

> Key25 moves back one: ATTACK→ZSSZBJ.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 19-28

Substitution ciphers
• Replaces one symbol with another.
• Categories monoalphabetic or polyalphabetic ciphers.
(1) Monoalphabetic Ciphers
o The relationship between a symbols in the plaintext to a symbol in the ciphertext is always one-to-one.
o sometimes called additive cipher or shift cipher or Caesar cipher (key of 3)
o When the cipher is additive, the plaintext, ciphertext, and key are integers in Z26.

Question sources

Computer Science model exam 2 · Question 34

Question 38 | Q01247

All are properties of a substitution cipher except:

A. The identity of the letters is changed in the encryption process.

B. The position of the letters is changed in the encryption process.

C. Uses the same key for the encryption and decryption process.

D. It has a secure key distribution problem.

Hint

Separate symbol replacement from position changes and apply the alphabet and modular direction.

Answer B: The position of the letters is changed in the encryption process.

Substitution replaces symbol identities; transposition changes positions (PDF p.6). Classical encryption is shared-key with a key-distribution problem (p.7), so position change is the exception in the standard pure-substitution model.

Why each choice is right or wrong

A - Incorrect: Changing letter identity is the defining substitution action. See PDF p.6, PDF p.7.

B - Correct: Changing letter positions defines transposition, not pure substitution. See PDF p.6, PDF p.7.

C - Incorrect: The module classifies classical encryption as shared-key encryption/decryption. See PDF p.6, PDF p.7.

D - Incorrect: The module explains the need to distribute the shared secret securely. See PDF p.6, PDF p.7.

Primary note proof

Study-note section SEC13

Substitution replaces symbols; transposition changes their positions. A monoalphabetic substitution preserves one mapping throughout the message and leaks frequency patterns despite a large nominal key space. Classical ciphers are teaching devices, not modern data protection.

Study-note section SEC16

Symmetric encryption uses a shared secret and efficiently protects bulk data.

Study-note section SEC16

Pairwise independent keys for n users require n(n-1)/2 shared keys; a trusted key distribution center changes that architecture.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 19-28

Substitution ciphers
• Replaces one symbol with another.
• Categories monoalphabetic or polyalphabetic ciphers.
(1) Monoalphabetic Ciphers
o The relationship between a symbols in the plaintext to a symbol in the ciphertext is always one-to-one.
o sometimes called additive cipher or shift cipher or Caesar cipher (key of 3)
o When the cipher is additive, the plaintext, ciphertext, and key are integers in Z26.

Computer Security module | PDF page 6 | lines 73-78

Transposition Ciphers
• Does not substitute one symbol for another, instead it changes the location of the symbols (reorders symbols).
• A transposition cipher relies on an algorithm which rearranges the order of the letters in a plaintext message

Computer Security module | PDF page 7 | lines 48-54

Symmetric key cryptography
• Uses conventional / secret-key / single-key.
• The secret key shared by both sender and receiver.
• All classical encryption algorithms are private-key.
o Use the same key (shared key) for encryption and decryption process.
• If this key is disclosed communications are compromised.

Computer Security module | PDF page 7 | lines 69-79

• 2 requirements for secure use of symmetric encryption:
o Strong encryption algorithm: i.e., Y = EK(X)
o Secret key known only to sender / receiver: i.e. X = DK(Y)
• Assume encryption algorithm is known
o Implies a secure channel to distribute key:
✓ Problem: - Key distribution problem
• Key Distribution
o Symmetric schemes require both parties to share a common secret key.
o Issue is how to securely distribute this key.

Question sources

Computer Science model exam 2 · Question 68

Question 39 | Q01282

Which statement is true? Assume a correctly used one-time pad and repeated-key Vigenère; compare the general cryptographic methods rather than one specially chosen implementation.

A. A correctly used one-time pad provides perfect secrecy, unlike repeated-key Vigenère

B. Symmetric key cryptography has a slow encryption process than asymmetric key encryption technique

C. Hill cipher remains secure even with enough independent known plaintext/ciphertext blocks

D. All of the above

Hint

Check the precise ordering or key-use assumptions rather than relying on a cipher's name.

Answer A: A correctly used one-time pad provides perfect secrecy, unlike repeated-key Vigenère

A one-time pad has perfect secrecy only under its strict randomness, secrecy, length and nonreuse conditions; repeated-key Vigenère leaks periodic structure. Symmetric encryption supplies efficient bulk protection, while public-key mechanisms commonly handle keys in hybrid designs, so B reverses the general efficiency account. Hill is linear: enough independent known blocks allow recovery of its key matrix, contradicting C. Because B and C are false, the all-of-the-above selection is false and A is correct.

Why each choice is right or wrong

A - Correct: Correct: the stated pad conditions provide perfect secrecy; repeated-key Vigenère has periodic structure.

B - Incorrect: Incorrect: symmetric encryption is the efficient bulk-data component; the proposed general slowdown reverses that role in hybrid protection.

C - Incorrect: Incorrect: Hill's linear rule permits recovery of K = C P^-1 mod 26 from sufficient independent known blocks.

D - Incorrect: Incorrect: B and C are false, so not all statements hold.

Primary note proof

Study-note section SEC15

A one-time pad XORs bits with an independent uniformly random secret key at least as long as the message and never reused. Its perfect secrecy depends on every condition.

Study-note section SEC14

The same plaintext letter may encrypt differently by position; repeated short keys still leak periodic structure.

Study-note section SEC14

It is linear: enough independent known plaintext/ciphertext blocks reveal K=CP^-1 mod26.

Study-note section SEC16

Symmetric encryption uses a shared secret and efficiently protects bulk data.

Study-note section SEC18

Hybrid systems establish or encapsulate a symmetric secret with public-key mechanisms, then protect bulk data symmetrically.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 63-72

o Vigenere Cipher
o Exercise:
✓ Encrypt the message —She is listening‖ using the 6-character keyword —PASCAL‖.
✓ The initial key stream is (15, 0, 18, 2, 0, 11). The key stream is the repetition of this initial key stream (as many times as needed)
o Hill Cipher:
✓ Key in the Hill cipher: The key matrix in the Hill cipher needs to have a multiplicative inverse.

Computer Security module | PDF page 8 | lines 4-20

• Given parties A and B have various key distribution alternatives:
o A can select key and physically deliver to B.
o Third party can select & deliver key to A & B.
o If A & B have communicated previously can use previous key to encrypt a new key.

o If A & B have secure communications with a third-party C, C can relay key between A & B.
• Key Hierarchy
o session key
✓ temporary key
✓ used for encryption of data between users
✓ for one logical session then discarded
o master key
✓ used to encrypt session keys
✓ shared by user & key distribution center
• Advantage: Simpler and faster

Question sources

Computer Science model exam 2 · Question 103

Question 40 | Q01509

An access control mechanism which works by comparing security labels with security clearances

A. Mandatory Access Control

B. Attribute Based Access Control

C. Discretionary Access Control

D. Role Based Access Control

Hint

Identify subject, object and permitted operation before comparing policy models.

Answer A: Mandatory Access Control

Security labels and clearances identify the mandatory policy mechanism in SEC09, elaborated for multilevel databases in SEC32. Owners cannot override that centrally enforced label policy. RBAC assigns role permissions and DAC allows owner grants; generic ABAC can evaluate attributes, but the specified clearance/classification mechanism is the named mandatory control model.

Why each choice is right or wrong

A - Correct: Mandatory control compares classifications/clearances under a policy ordinary owners cannot override.

B - Incorrect: ABAC evaluates general attributes; the named security-label/clearance mechanism specifically identifies mandatory control.

C - Incorrect: DAC allows owners to grant permissions rather than enforcing nonoverridable classification labels.

D - Incorrect: RBAC bases permissions on roles and user-role assignments rather than security-label clearance comparisons.

Primary note proof

Study-note section SEC09

Subjects request operations on objects. An ACL associates an object/resource with allowed subjects or actions; a network ACL matches traffic such as address, protocol and port. Classification and filtering are direct effects; high availability or malware detection is not guaranteed by an ACL.

Study-note section SEC09

DAC lets owners grant permissions; mandatory control enforces labels and policy that ordinary owners cannot override.

Study-note section SEC32

A multilevel database stores items at different labels and enforces mandatory policy using subject clearances, classifications and compartments.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 80-87

6. Access control
• Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Computer Security module | PDF page 11 | lines 88-100

• Network access control (NAC): is an umbrella term for managing access to a net-work. NAC authenticates users logging into the network and determines what data they can access and actions they can perform. NAC also examines the health of the user’s computer or mobile device (the endpoints).
• An operating system provides an access enforcement mechanism.
• Two fundamental concepts of access control:
o a protection system that defines the access control specification and
o a reference monitor that is the system’s access enforcement mechanism that enforces this specification.

Question sources

Mock exam 1 · Question 25

Question 41 | Q01510

An attack in which the DNS queries are wrongly resolved in order to redirect the users to a malicious website.

A. DNS Flooding attack

B. NXDomain attack

C. SNMP Reflection attack

D. DNS Hijacking attack

Hint

Identify the trust claim or communication mapping being abused.

Answer D: DNS hijacking/spoofed resolution

Wrong name-to-address answers redirecting users match falsified DNS resolution, described as spoofing/cache poisoning in SEC05 and called hijacking in D. DNS and NXDOMAIN floods instead stress query capacity; they need not provide a malicious replacement destination. No SNMP reflection mechanism is described by the stem.

Why each choice is right or wrong

A - Incorrect: DNS flooding exhausts query-processing capacity rather than specifically supplying false destination records.

B - Incorrect: NXDOMAIN attacks generate nonexistent-name queries to stress resolvers, not the false successful answers described.

C - Incorrect: The stem describes DNS answer manipulation, not reflected traffic using a different management protocol.

D - Correct: Hijacking DNS resolution redirects a name to an attacker-controlled destination through false resolution data.

Primary note proof

Study-note section SEC05

> Phishing impersonates a trusted source to obtain secrets or induce an unsafe action; spear phishing targets a specific person. Verify a request through an independently known channel. Shoulder surfing observes entry; pretexting invents a reason for disclosure. Monitoring software used with authorization is not necessarily an attack merely because it records activity.

Study-note section SEC06

> DNS flooding targets query capacity; NXDOMAIN attacks generate queries for nonexistent names to stress resolvers.

Study-note section SEC05

> ARP spoofing poisons local IPv4-to-MAC associations; DNS spoofing/cache poisoning supplies false name resolutions; IP spoofing falsifies a source address.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 42-47

> • Denial of Service (DoS) Attack: Blocking access from legitimate users
> • Distributed DoS Attack: is accomplished by tricking routers into attacking a target or using Zombie hosts to simultaneously attack a given target with large number of packets.

Computer Security module | PDF page 5 | lines 65-74

> • Reconnaissance: The hackers first identifies a target to launch an attack, extract maximum information regarding this target, understand its vulnerabilities, and then only explore the best ways to exploit it.
> o Examples: Credential Theft using Keyloggers and Spyware; Identity Theft using spam e-mail, phishing attack and social engineering attack
> • Access: Ones the attacker Reconnaissance the target, S/he starts a successful exploitation and continued access to the system. Then, the attacker is in control of the target.

Question sources

Mock exam 1 · Question 26

Question 42 | Q01511

The term injection attack refers to a wide variety of program flaws related to invalid handling of input data. Which one of the following is not a type of injection attack?

A. Cross site scripting

B. Input fuzzing

C. SQL injection

D. Command injection

Hint

Locate where untrusted data can become instructions and where they execute.

Answer B: Input fuzzing is testing, not an injection-attack category

SEC26 directly distinguishes fuzzing from injection: fuzzing varies unexpected inputs to discover defects. SQL injection changes SQL syntax and XSS introduces attacker-controlled script into a browser context. Command injection likewise names unsafe interpretation of input as commands. Fuzzing may discover those defects but does not itself identify that attack mechanism.

Why each choice is right or wrong

A - Incorrect: XSS injects attacker-controlled script/content into a browser execution context, so it is an injection-type attack.

B - Correct: Input fuzzing is a technique for finding faults with varied unexpected inputs, not an injection category.

C - Incorrect: SQL injection explicitly alters query syntax through unsafe handling of untrusted values.

D - Incorrect: Command injection names untrusted input interpreted as executable commands rather than merely a testing technique.

Primary note proof

Study-note section SEC26

> SQL injection occurs when untrusted values alter SQL syntax through unsafe string construction. A tautology can change which rows match. Bind data values in prepared statements, allow-list identifiers that cannot be parameters, and restrict the application's database account. Stored procedures/ORMs are safe only when they avoid unsafe dynamic SQL.

Study-note section SEC26

> Fuzzing sends unexpected inputs to discover faults; it is a testing technique, not itself a category of injection.

Study-note section SEC-S01

> This is an injection flaw in command interpretation, unlike fuzzing, which supplies varied inputs to discover defects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 71-73

> o Database Based :
> ✓ identify the security exposure in the database
> systems to prevent from SQL Injections.

Computer Security module | PDF page 11 | lines 12-16

> • Secure Web servers:
> o The main vulnerability of web applications is Cross-Site
> Scripting (XSS).
> o Securing the website or web application itself and the
> network around it.

Question sources

Mock exam 1 · Question 27

Mock exam 2 · Question 22

Question 43 | Q01590

Using the convention prover-to-verifier, which statement correctly describes the direction of an authentication setting?

A. Host-to-host authentication allows the host to prove the identity of a user

B. User-to-user authentication allows the user to prove the identity of a machine

C. User-to-host authentication allows the user to prove the identity of a machine

D. Host-to-user authentication lets the host prove its identity to the user

Hint

Separate identity claims from proof, and factor type from factor count.

Answer D: Host-to-user authentication lets the host prove its identity to the user

The arrow convention identifies the entity supplying identity proof first and the verifier second. A user-to-host exchange therefore proves the user's identity to the host, while host-to-user proves the host's identity to the user. The corrected final option follows that rule. The other options describe proving a different entity's identity from the one indicated as prover, so they do not match their stated authentication setting.

Why each choice is right or wrong

A - Incorrect: Host-to-host concerns a host proving identity to another host, not proving a user identity.

B - Incorrect: User-to-user concerns a user proving identity to another user, not a machine.

C - Incorrect: User-to-host proves the user's identity, not the machine's.

D - Correct: The host is the prover and the user is the verifier, as the convention requires.

Primary note proof

Study-note section SEC07

> A common access sequence is identify the subject, authenticate the claim, authorize the operation, then record relevant events. Identification is the username/claim, not its proof. When a sequence begins with authentication, distinguish it from the preceding identification stage. User-to-host means a user proves identity to a host; host-to-user means the host proves its identity to a user; the arrow names the prover and verifier.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 15-27

> • Authentication
> o is proving a claim - usually that you are who you say you are.
> o for example it may be obtained by the provision of a

password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Computer Security module | PDF page 11 | lines 74-79

• Most OS authentication is handled through user names and passwords. Biometric (e.g. voice, face, retina, iris, fingerprint) and physical token-based (swipe cards, pin-generating cards) authentication are sometimes used to augment simple passwords, but the costs and accuracy of the technology limit their adoption.

Question sources

Mock exam 3 · Question 22

Question 44 | Q01591

Which one of the following is not a correct match?

A. Biba Model - Integrity

B. Bell-LaPadula Model - Confidentiality

C. Clark-Wilson Model - Confidentiality

D. Multiple Levels of Security (MLS) Model - Confidentiality

Hint

Apply the named model's read and write restrictions separately.

Answer C: Clark-Wilson protects integrity, not confidentiality

SEC10 directly gives Biba and Clark-Wilson integrity goals, and Bell-LaPadula confidentiality. SEC32 describes multilevel label enforcement for confidentiality. C therefore mismatches the model and goal; the others are compatible with the note's defined model purposes.

Why each choice is right or wrong

A - Incorrect: Biba is an integrity model, so this pairing is correct rather than the requested mismatch.

B - Incorrect: Bell-LaPadula protects confidentiality, making this pairing correct.

C - Correct: Clark-Wilson enforces integrity through well-formed transformations and separation of duties, not confidentiality.

D - Incorrect: Multilevel label restrictions can protect confidentiality, as SEC32 explains.

Primary note proof

Study-note section SEC10

> Bell-LaPadula protects confidentiality. The simple-security rule prohibits reading above a subject's level; the star rule prohibits writing down. The strong-star variant permits writing only at the same level. With TS>S>C>U, an S subject can read C under the simple rule but cannot write C under star; compartments and discretionary restrictions may still deny access.

Study-note section SEC10

> Clark-Wilson protects integrity through certified well-formed transformations, integrity checks and separation of duties.

Study-note section SEC10

> It is not a confidentiality model. "MAC" is an enforcement category and does not force every system to use Bell-LaPadula rather than another label policy.

Related reading (context, not primary note proof)

Study-note section SEC32

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 79-86

> • Integrity: covers Data integrity and System integrity
> o the prevention of unauthorized writing or modification of information.
> o Integrity in a computer system means that there is an external consistency in the system - everything is as it is expected to be.
> o Data integrity: the data stored on the computer is the same as what is intended.

Computer Security module | PDF page 3 | lines 71-78

> • Confidentiality: Data confidentiality and Privacy
> o is the prevention of unauthorized disclosure of information.
> o In other words, confidentiality means keeping information private or safe.
> o it may be important for military, business or personal reasons.
> o it may also be known as privacy or secrecy.

Computer Security module | PDF page 11 | lines 80-87

6. Access control
• Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Question sources

Mock exam 3 · Question 23

Question 45 | Q01634

What is the first step of access control?

A. Subject identification

B. ACL verification

C. Subject authorization

D. Accountability logging

Hint

Separate identity claims from proof, and factor type from factor count.

Answer A: Subject identification

PDF p.4 says users must be identified and authenticated for access; p.11 then describes authorization of operations. Identification supplies the claimed subject before checking identity and permissions.

Why each choice is right or wrong

A - Correct: Subject identification supplies the identity on which subsequent checks operate. See PDF p.4, PDF p.11.

B - Incorrect: ACL checking requires knowing the subject and request first. See PDF p.4, PDF p.11.

C - Incorrect: Authorization decides permitted operations after the identity is established. See PDF p.4, PDF p.11.

D - Incorrect: Accountability records actions and does not establish the initial identity. See PDF p.4, PDF p.11.

Primary note proof

Study-note section SEC07

A common access sequence is identify the subject, authenticate the claim, authorize the operation, then record relevant events. Identification is the username/claim, not its proof. When a sequence begins with authentication, distinguish it from the preceding identification stage.

User-to-host means a user proves identity to a host; host-to-user means the host proves its identity to a user; the arrow names the prover and verifier.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Computer Security module | PDF page 11 | lines 88-100

• Network access control (NAC): is an umbrella term for managing access to a net-work. NAC authenticates users logging into the network and determines what data they can access and actions they can perform. NAC also examines the health of the user's computer or mobile device (the endpoints).
• An operating system provides an access enforcement mechanism.
• Two fundamental concepts of access control:
o a protection system that defines the access control specification and
o a reference monitor that is the system's access enforcement mechanism that enforces this specification.

Question sources

Mock exam 2 · Question 23

Question 46 | Q01635

Which of the following is not a reason why using passwords alone is a poor security mechanism?

A. Short passwords can be discovered quickly in brute force attacks only when used against a stolen password database file.

B. When possible, users choose easy-to-remember passwords, which are therefore easy to guess or crack.

C. Passwords can be stolen through many means, including observation, recording and playback, and security database theft.

D. Randomly generated passwords are hard to remember, thus many users write them down.

Hint

Consider what the attacker can guess or steal, or which component validates a claim.

Answer A: The “only” restriction to stolen-database brute force is false

The negative stem asks which offered reason is not valid. A incorrectly restricts brute-force exposure of short passwords to stolen database files. SEC08 explicitly permits failure in online as well as offline settings, subject to rate limits. Guessable choices, theft and insecure recording are legitimate password-only weaknesses rather than exclusive claims.

Why each choice is right or wrong

A - Correct: Short passwords can be attacked online too; only against a stolen database is an unjustified restriction.

B - Incorrect: Predictable memorable choices enable likely-word guessing, a real weakness of password-only security.

C - Incorrect: Observed entry and stolen verifiers are real exposure paths, consistent with the note’s password and monitoring threats.

D - Incorrect: Difficulty remembering random secrets can encourage unsafe recording; password managers address that operational weakness.

Primary note proof

Study-note section SEC08

> Brute force enumerates guesses; dictionary attacks prioritize likely words; credential stuffing tries previously leaked credentials on another service. Online attempts face server rate limits; offline attacks against stolen verifiers do not. Short passwords can fail in either setting. Use unique long secrets and password managers, rate limiting and suitable MFA; store salted slow password hashes, not reversible plaintext passwords. A salt prevents shared precomputed lookup, but is not secret.

Study-note section SEC05

> Shoulder surfing observes entry; pretexting invents a reason for disclosure.

Study-note section SEC05

ARP spoofing poisons local IPv4-to-MAC associations; DNS spoofing/cache poisoning supplies false name resolutions; IP spoofing falsifies a source address.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 74-79

• Most OS authentication is handled through user names and passwords. Biometric (e.g. voice, face, retina, iris, fingerprint) and physical token-based (swipe cards, pin-generating cards) authentication are sometimes used to augment simple passwords, but the costs and accuracy of the technology limit their adoption.

Computer Security module | PDF page 5 | lines 50-60

• Password attacks: guessing password; social engineering attack, Dictionary attack
• Cryptographic Attack:
A. Frequency Analysis attack: Guess values based on frequency of occurrence
B. Brute-force attack: The attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried to achieve success.
C. Cryptanalysis: the process of breaking an encrypted code.

Computer Security module | PDF page 5 | lines 65-74

• Reconnaissance: The hackers first identifies a target to launch an attack, extract maximum information regarding this target, understand its vulnerabilities, and then only explore the best ways to exploit it.
o Examples: Credential Theft using Keyloggers and Spyware; Identity Theft using spam e-mail, phishing attack and social engineering attack
• Access: Ones the attacker Reconnaissance the target, S/he starts a successful exploitation and continued access to the system. Then, the attacker is in control of the target.

Question sources

Mock exam 2 · Question 24

Question 47 | SM0040

Which one of the following is a measure taken to ensure information and information systems' availability, integrity, authentication and confidentiality?

A. Wiretapping

B. Information security

C. Information assurance

D. Interception

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer C: Information assurance

PDF p.2 explicitly defines IA as measures protecting information and systems by ensuring availability, integrity, authentication, confidentiality and non-repudiation. This matches the stem's list most closely.

Why each choice is right or wrong

A - Incorrect: Wiretapping is an interception behavior, not an assurance measure. See PDF p.2.

B - Incorrect: InfoSec is related but the exact enumerated measures definition is assigned to IA. See PDF p.2.

C - Correct: Information assurance matches the module's definition almost verbatim. See PDF p.2.

D - Incorrect: Interception compromises confidentiality rather than ensuring these properties. See PDF p.2.

Primary note proof

Study-note section SEC01

> Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC01

> Non-repudiation supplies evidence supporting origin or receipt; signatures do not make denial physically impossible.

Study-note section SEC01

> Information assurance includes confidence, risk management, protection, detection, response and restoration across the information lifecycle.

Related reading (context, not primary note proof)

Study-note section SEC03

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 2 | lines 77-83

> • IA: measures that protect and defend information and ISs by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation.
> • Information protection or InfoSec: protection of information and ISs from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction.

Question sources

MOE Computer Science Exit Exam · Question 12-2

National Computer Science Exit Exam · Question 37

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 166

Question 48 | SM0044

Which one of the following is a security service enforced to protect disclosure of information (stored in file or in transmission) from unauthorized entities?

A. Authentication

B. Integrity

C. Availability

D. Confidentiality

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer D: Confidentiality

Unauthorized disclosure of stored or transmitted information is precisely the confidentiality concern (PDF p.3), independent of storage location.

Why each choice is right or wrong

A - Incorrect: Authentication proves identity, not disclosure restrictions. See PDF p.3, PDF p.4.

B - Incorrect: Integrity protects against unauthorized changes. See PDF p.3, PDF p.4.

C - Incorrect: Availability protects timely access. See PDF p.3, PDF p.4.

D - Correct: Confidentiality prevents unauthorized disclosure. See PDF p.3, PDF p.4.

Primary note proof

Study-note section SEC01

> Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC01

> Data security protects information at rest, in use and in transit, including accidental corruption as well as intentional attacks.

Study-note section SEC01

> Authentication establishes an identity or other claim; authorization decides permitted actions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 71-78

> • Confidentiality: Data confidentiality and Privacy
> o is the prevention of unauthorized disclosure of information.
> o In other words, confidentiality means keeping information private or safe.
> o it may be important for military, business or personal reasons.
> o it may also be known as privacy or secrecy.

Computer Security module | PDF page 4 | lines 15-27

> • Authentication
> o is proving a claim - usually that you are who you say you are.
> o for example it may be obtained by the provision of a password or by a scan of your retina.
> • Access controls
> o provide the limitation and control of access to authorized users through identification and authentication.
> o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
> o In a large system there may be a complex structure

determining which users and applications have access to which objects.

Computer Security module | PDF page 3 | lines 79-86

• Integrity: covers Data integrity and System integrity
o the prevention of unauthorized writing or modification of information.
o Integrity in a computer system means that there is an external consistency in the system - everything is as it is expected to be.
o Data integrity: the data stored on the computer is the same as what is intended.

Computer Security module | PDF page 3 | lines 87-91

• Availability:
o Assures that systems work promptly and service is not denied to authorized users.
o Information should be accessible and usable upon appropriate demand by an authorized user.

Question sources

MOE Computer Science Exit Exam · Question 13-2

National Computer Science Exit Exam · Question 52

National Computer Science Exit Exam · Question 57

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 167

Question 49 | SM0054

Which general security-control category creates a protected network boundary by enforcing allow/block traffic rules for a group of machines?

A. Network operating system

B. Firewall

C. Router used only for forwarding, without security filtering

D. Switch used only for forwarding, without security filtering

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer B: Firewall

A network firewall protects a group of machines by enforcing traffic policy at a boundary. The decisive function is allowing or blocking traffic under security rules. A network operating system is not the name of that specific policy-enforcement role. The other two alternatives are explicitly limited to forwarding without security filtering, so their connectivity functions do not meet the requirement. Actual routers, switches or proxies can include firewall capabilities; the stated scope prevents that implementation overlap from making the choices ambiguous.

Why each choice is right or wrong

A - Incorrect: Incorrect: a network operating system does not name the specific boundary traffic-policy function requested.

B - Correct: Correct: a firewall enforces the allow/block rules protecting the group.

C - Incorrect: Incorrect: the explicitly forwarding-only router does not perform the required security filtering.

D - Incorrect: Incorrect: the explicitly forwarding-only switch does not perform the required security filtering.

Primary note proof

Study-note section SEC23

> A firewall enforces traffic policy.

Study-note section SEC23

> Host firewalls protect individual machines; network firewalls segment networks and can create a DMZ for public-facing services.

Study-note section SEC-S03

> A router forwards IP packets between networks; forwarding alone is different from detecting suspicious behavior or enforcing a firewall's security policy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 22-34

> 3. Firewall
> • a program or network devices (e.g. router) which filters
> access to a protected network from the internet connection.
> • protects a local or network of systems from network-based
> security threats,
> • monitors & controls incoming & outgoing network traffic
> based on predetermined security rules.
> • Firewall Objectives
> o Keep intruders, malicious code and unwanted traffic
> out
> o Keep private and sensitive information in

o security wall b/n private (protected) NW & outside word.

Computer Security module | PDF page 11 | lines 35-56

• Categorizes of firewalls
a) Network firewalls: Filter traffic b/n two or more networks and run-on NW hardware.
✓ Implemented at a specific point in the network path and protects all computers on the internal side of the firewall from all computers on the external side of the fire wall.
b) Host-based firewalls: Run on host computers and control network traffic in/out of those machines.
a) Packet Filtering Router: Examines TCP/IP header information of network data packets going in both directions. Applies set of rules to each incoming IP packets & then forwards or discards the packets.
b) Application-Level Gateway (application firewall / proxy server): runs special software that acts as a proxy for a service request. Proxy server receives requests for Web pages, accesses the Web server on behalf of the external client, and returns the requested pages to the users.
c)Circuit Level Gateway: Standalone Software that sets up two TCP connections (inside & outside). It determines which connections will be allowed.

Question sources

MOE Computer Science Exit Exam · Question 16-1

National Computer Science Exit Exam · Question 86

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 168

Question 50 | SM0058

Which one of the following best explains attackers who are motivated by financial gain and perform sabotage, IP theft and steal secrets of a rival company?

A. Stealer

B. Competitor

C. Hacktivist

D. Activist

Hint

Classify the action actually described and distinguish observation, alteration, disruption and identity claims.

Answer B: Competitor

The rival-company context and commercial-secret theft identify a competitor's motive, as SEC03 states. Financial gain alone could describe many attackers, but the specific rival-company relationship distinguishes B. Hacktivists/activists pursue a cause, while stealer is a generic activity label rather than the specified commercial relationship.

Why each choice is right or wrong

A - Incorrect: Stealer describes taking information but does not identify the rival-company commercial motive.

B - Correct: A competitor can seek commercial secrets and advantage over a rival company.

C - Incorrect: Hacktivist motivation is a cause rather than the specified commercial competition.

D - Incorrect: An activist pursues a cause; the scenario instead identifies a financially motivated rival-company interest.

Primary note proof

Study-note section SEC03

> White-hat testing requires authorization and scope; black-hat activity is malicious or unauthorized. A competitor may seek commercial secrets; hacktivists pursue a cause. "Hacker" alone does not establish legality. Insider versus outsider describes origin/trust position, not the specific mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 50-61

> • Types of Attacks on Data CIA (W. Stallings)
> o Disclosure/Interception: unauthorized party snooping, or getting access to a resource, Attack on data confidentiality:
> o Modification / Deception: unauthorized party modifying a resource, acceptance of false data, E.g., providing wrong data (attack on data integrity)
> o Disruption/Interruption: destroyed/unavailable services/resources, DoS (attack on data availability)
> o Fabrication: unauthorized party inserts a fake asset/resource, loss of confidentiality, authenticity and integrity of the message

Question sources

MOE Computer Science Exit Exam · Question 17-1

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 169

Question 51 | SM0060

Against an attacker whose only access is direct communication over the public Internet, which computer removes that attack path? Assume the disconnected computer has no Internet-reachable intermediary or other network route to the attacker.

A. A computer connected to the Internet with the latest firewall system

B. A computer connected to the Internet having strong intrusion detection

C. A computer that is not connected to the Internet

D. A computer connected to the Internet and the latest anti-virus installed on it

Hint

Start with the stated attacker path or physical hazard and the limits of each control.

Answer C: A computer that is not connected to the Internet

Disconnecting removes the Internet communication path available to the specified attacker. Firewalls, intrusion detection and antivirus can reduce particular risks, but they do not remove the fact that the other choices remain connected. The conclusion is deliberately limited to that attack path: the disconnected system can still face insiders, removable media or physical threats. Those are excluded from this scenario rather than silently treated as impossible.

Why each choice is right or wrong

A - Incorrect: A firewall-equipped Internet connection still exposes an Internet communication path.

B - Incorrect: Intrusion detection observes activity but does not remove connectivity.

C - Correct: The stated disconnected system has no path available to this attacker.

D - Incorrect: Antivirus does not remove the Internet communication path.

Primary note proof

Study-note section SEC11

> Protect equipment against theft, unauthorized connections, heat, humidity, water, electrical surges and fire. A UPS supplies temporary conditioned power; it is normally protective, but faulty installation/battery failure can itself be hazardous. A disconnected machine loses many network attack paths yet remains exposed to insiders, removable media, supply-chain attacks and physical access. There is no universal “most secure computer” without a threat model.

Study-note section SEC23

> Bypass paths, insiders, encrypted uninspected content and infected removable media limit firewall protection.

Study-note section SEC23

Host firewalls protect individual machines; network firewalls segment networks and can create a DMZ for public-facing services.

Related reading (context, not primary note proof)

Study-note section SEC24

Study-note section SEC04

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 65-68

4. Physical security
• Without physical security to protect hardware (i.e., doors that lock) nothing else about a computer system can be called secure

Computer Security module | PDF page 11 | lines 22-34

3. Firewall
• a program or network devices (e.g. router) which filters access to a protected network from the internet connection.
• protects a local or network of systems from network-based security threats,
• monitors & controls incoming & outgoing network traffic based on predetermined security rules.
• Firewall Objectives
o Keep intruders, malicious code and unwanted traffic out
o Keep private and sensitive information in
o security wall b/n private (protected) NW & outside word.

Computer Security module | PDF page 5 | lines 2-10

• Principles of Least Privilege: —Do not give any more privileges than absolutely necessary to do/perform the required job‖.
• Defense in Depth: having more than one layer or type of defense
• Minimization: —do not run any software, or applications that are not strictly required to do the entrusted job.‖
• Compartmentalization: limits the damage in other compartments when one compartment is compromised.

Question sources

MOE Computer Science Exit Exam · Question 17-3

National Computer Science Exit Exam · Question 57

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 170

Question 52 | SM0072

Suppose you have received a message from your friend but the message is scrambled for security. Your friend used a program to encode messages into scrambled text. In order to read the message,
which program would you use?

A. Encryption program

B. Ciphertext

C. Secret key

D. Decryption program

Hint

Trace message transformations and separate data, keys, algorithms and codebreaking.

Answer D: Decryption program

The received message is ciphertext; recovering its readable plaintext is deciphering/decryption on PDF p.6. The question asks for the program carrying out that inverse operation.

Why each choice is right or wrong

A - Incorrect: Encryption converts plaintext into ciphertext, the opposite direction. See PDF p.6.

B - Incorrect: Ciphertext is data, not the recovery program. See PDF p.6.

C - Incorrect: A secret key may be an input but is not a program. See PDF p.6.

D - Correct: A decryption program performs plaintext recovery. See PDF p.6.

Primary note proof

Study-note section SEC12

> Plaintext is the original message, ciphertext its encrypted representation, a key is input to an algorithm, encryption computes C=E_k(P), and decryption recovers P=D_k(C). The five conventional symmetric ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. Cryptanalysis seeks recovery or other breaks without the authorized secret; cryptology includes cryptography and cryptanalysis. Steganography hides that a message exists; encryption hides its meaning.

Study-note section SEC12

> Security depends on algorithm, key size, randomness, mode and implementation.

Study-note section SEC12

Assume the algorithm is known and protect keys: secrecy of implementation is not the main security argument.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 2-12

o Encipher (encrypt) - the process of converting plaintext to ciphertext
o Decipher (decrypt) - recovering plaintext from ciphertext
o Cryptography - the study of encryption principles/methods
o Cryptanalysis (codebreaking) - the study of principles/ methods of breaking ciphertext without knowing key
o Cryptology - the field of study which deals with both cryptography and cryptanalysis

Question sources

MOE Computer Science Exit Exam · Question 20-4

National Computer Science Exit Exam · Question 6

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 171

Question 53 | SM0081

Which network security issue shows the level of weakness in networks and devices?

A. Attack

B. Availability

C. Vulnerability

D. Treat

Hint

Distinguish weaknesses, potential harmful events, actual attempts and the effect of a control.

Answer C: Vulnerability

A vulnerability is an exploitable weakness or fault in an information system or component (PDF p.3). The question asks about weakness, not the attack event or service goal.

Why each choice is right or wrong

A - Incorrect: Attack is an actual security violation. See PDF p.3.

B - Incorrect: Availability is a goal of usable access. See PDF p.3.

C - Correct: Vulnerability is the defined weakness. See PDF p.3.

D - Incorrect: Treat appears to mean threat, a potential harmful circumstance rather than the weakness. See PDF p.3.

Primary note proof

Study-note section SEC02

An asset is something valuable: data, software, hardware, service or reputation. A threat is a potentially harmful circumstance; a vulnerability is an exploitable weakness; an attack is an actual attempt; a control reduces likelihood or impact. A software flaw is not automatically an active attack. Risk concerns likelihood and consequence, not a universally meaningful numeric product of two unmeasured labels.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 24-40

• Threat: any circumstances or events that can potentially harm an information system by destroying it, disclosing the information stored on the system, adversely modifying data, or making the system unavailable.
• Vulnerability: weaknesses or fault in an information system or its components that could be exploited.
• Attack: actual violation of security that derives from an intelligent threat.
• Risk: An expectation of loss expressed as the probability that a particular threat will exploit a particular vulnerability with a particular harmful result.
o Risk = Threats x Vulnerabilities
• Control, safeguard, or countermeasure: An action, device, procedure, or technique that reduces a threat, a vulnerability, or an attack by eliminating or preventing it, by minimizing the harm it can cause, or by discovering and reporting it so that corrective action can be taken.

Question sources

MOE Computer Science Exit Exam · Question 23-2

National Computer Science Exit Exam · Question 33

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 172

Question 54 | SM0096

Which one is a method used to obtain the meaning of encrypted information without access to the secret key?

A. Cryptography

B. Crypto-analysis

C. Decryption

D. Encryption

Hint

Trace message transformations and separate data, keys, algorithms and codebreaking.

Answer B: Crypto-analysis

Obtaining meaning without the secret key matches cryptanalysis, which the module explicitly contrasts with ordinary decryption (PDF p.6).

Why each choice is right or wrong

A - Incorrect: Cryptography studies encryption methods rather than without-key breaking. See PDF p.6.

B - Correct: Crypto-analysis matches breaking ciphertext without knowing the key. See PDF p.6.

C - Incorrect: Decryption describes plaintext recovery but lacks the no-key distinction. See PDF p.6.

D - Incorrect: Encryption produces ciphertext rather than recovering meaning. See PDF p.6.

Primary note proof

Study-note section SEC12

> Plaintext is the original message, ciphertext its encrypted representation, a key is input to an algorithm, encryption computes C=E_k(P), and decryption recovers P=D_k(C). The five conventional symmetric ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. Cryptanalysis seeks recovery or other breaks without the authorized secret; cryptology includes cryptography and cryptanalysis. Steganography hides that a message exists; encryption hides its meaning.

Study-note section SEC12

> Symmetric cryptography is not inherently less secure than public-key cryptography, and speed ratios are not fixed constants.

Study-note section SEC12

> Security depends on algorithm, key size, randomness, mode and implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 2-12

o Encipher (encrypt) - the process of converting plaintext to ciphertext
o Decipher (decrypt) - recovering plaintext from ciphertext
o Cryptography - the study of encryption principles/methods
o Cryptanalysis (codebreaking) - the study of principles/ methods of breaking ciphertext without knowing key
o Cryptology - the field of study which deals with both cryptography and cryptanalysis

Question sources

MOE Computer Science Exit Exam · Question 27-3

National Computer Science Exit Exam · Question 66

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 173

Question 55 | SM0221

Which one of the following IS NOT a concern of availability?

A. Fault Tolerance

B. Alteration of resources

C. Usefulness of resources

D. Fair allocation of resources

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer B: Alteration of resources

Availability means timely, reliable, usable access for authorized users (PDF p.3). Altering resources is instead the integrity issue; fault tolerance, usability and fair allocation can support continued access.

Why each choice is right or wrong

A - Incorrect: Fault tolerance supports reliable service, consistent with availability. See PDF p.3.

B - Correct: Unauthorized alteration belongs to integrity rather than the access/continuity goal. See PDF p.3.

C - Incorrect: Useful/usable resources are expressly part of availability. See PDF p.3.

D - Incorrect: Fair allocation supports access for authorized users rather than denying service. See PDF p.3.

Primary note proof

Study-note section SEC01

> Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC01

> Accountability links actions to responsible entities through reliable identity and protected logs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 87-91

> • Availability:
> o Assures that systems work promptly and service is not denied to authorized users.
> o Information should be accessible and usable upon appropriate demand by an authorized user.

Computer Security module | PDF page 3 | lines 79-86

> • Integrity: covers Data integrity and System integrity
> o the prevention of unauthorized writing or modification of information.
> o Integrity in a computer system means that there is an external consistency in the system - everything is as it is expected to be.
> o Data integrity: the data stored on the computer is the same as what is intended.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 7-2

Computer Science Exit Exam practice collection · Question 180

Question 56 | SM0223

Person A sends a message to person B. While the message is transmitted, person C gains access to the message. Which category of attack is performed by person C?

A. Modification

B. Interception

C. Interruption

D. Fabrication

Hint

Classify the action actually described and distinguish observation, alteration, disruption and identity claims.

Answer B: Interception

C gains unauthorized access to the message in transit without any stated alteration, destruction or insertion. PDF p.4 calls this disclosure/interception.

Why each choice is right or wrong

A - Incorrect: No message change is specified. See PDF p.4.

B - Correct: Unauthorized message access matches interception. See PDF p.4.

C - Incorrect: No loss of service is specified. See PDF p.4.

D - Incorrect: No fake message is inserted. See PDF p.4.

Primary note proof

Study-note section SEC03

> Interception copies or observes data, modification alters it, interruption blocks it, and fabrication inserts false material. The 4D classification uses disclosure, deception, disruption and usurpation, the last meaning unauthorized control. DoS primarily demonstrates disruption/availability loss; it need not give control of the victim.

Study-note section SEC03

> Neither category is universally easy to prevent or detect.

Study-note section SEC03

> Passive attacks observe without changing the target exchange: sniffing and traffic analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 50-61

> • Types of Attacks on Data CIA (W. Stallings)
> o Disclosure/Interception: unauthorized party snooping,
> or getting access to a resource, Attack on data

confidentiality:
o Modification / Deception: unauthorized party modifying a resource, acceptance of false data, E.g., providing wrong data (attack on data integrity)
o Disruption/Interruption: destroyed/unavailable services/resources, DoS (attack on data availability)
o Fabrication: unauthorized party inserts a fake asset/resource, loss of confidentiality, authenticity and integrity of the message

Question sources

Computer Science Exit Exam · Sene 2016 · Question 8-1

Computer Science Exit Exam practice collection · Question 181

Question 57 | SM0227

Which one of the following terms refers to the techniques used for deciphering ciphertext without any knowledge of the enciphering mechanism?

A. Cryptography

B. Cryptology

C. Cryptoanalysis

D. Decryption

Hint

Trace message transformations and separate data, keys, algorithms and codebreaking.

Answer C: Cryptoanalysis

The qualifier without knowledge distinguishes cryptanalysis from ordinary deciphering in PDF p.6. Cryptology includes both protective methods and codebreaking, but is not the specific technique requested.

Why each choice is right or wrong

A - Incorrect: Cryptography studies encryption principles. See PDF p.6.

B - Incorrect: Cryptology is the umbrella field including cryptography and cryptanalysis. See PDF p.6.

C - Correct: Cryptoanalysis is the offered spelling of the without-key codebreaking concept. See PDF p.6.

D - Incorrect: Decryption is ordinary plaintext recovery, not specifically without secret knowledge. See PDF p.6.

Primary note proof

Study-note section SEC12

Plaintext is the original message, ciphertext its encrypted representation, a key is input to an algorithm, encryption computes C=E_k(P), and decryption recovers P=D_k(C). The five conventional symmetric ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. Cryptanalysis seeks recovery or other breaks without the authorized secret; cryptology includes cryptography and cryptanalysis. Steganography hides that a message exists; encryption hides its meaning.

Study-note section SEC12

Symmetric cryptography is not inherently less secure than public-key cryptography, and speed ratios are not fixed constants.

Study-note section SEC12

Security depends on algorithm, key size, randomness, mode and implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 2-12

o Encipher (encrypt) - the process of converting
plaintext to ciphertext
o Decipher (decrypt) - recovering plaintext from
ciphertext
o Cryptography - the study of encryption
principles/methods
o Cryptanalysis (codebreaking) - the study of
principles/ methods of breaking ciphertext without
knowing key
o Cryptology - the field of study which deals with both
cryptography and cryptanalysis

Question sources

Computer Science Exit Exam · Sene 2016 · Question 9-2

Computer Science Exit Exam practice collection · Question 182

Question 58 | SM0258

In asymmetric cryptography, the message is decrypted using _____.

A. Receiver's public key

B. Receiver's private key

C. Sender's private key

D. Sender's public key

Hint

Identify the intended recipient or signer and separate confidentiality from verification.

Answer B: Receiver's private key

For confidential public-key transmission, encryption uses the receiver's public key and decryption the receiver's private key (PDF p.8). The answer follows that communication model.

Why each choice is right or wrong

A - Incorrect: Receiver's public key performs encryption in this model. See PDF p.8.

B - Correct: Receiver's private key performs decryption. See PDF p.8.

C - Incorrect: Sender's private key is not the recipient decryption key. See PDF p.8.

D - Incorrect: Sender's public key belongs to signature verification, not confidential recipient decryption. See PDF p.8.

Primary note proof

Study-note section SEC18

> For confidentiality to B, encrypt with B's authenticated public encryption key; B decrypts with B's private key. Do not encrypt to A's public key if only B should read it, and never assume A possesses B's private key. For a signature, A signs with A's private signing key and others verify with A's public key. These are distinct operations; not every public-key algorithm supports encryption.

Study-note section SEC18

> For one-way public-key confidentiality the sender need not own a key pair; it needs the recipient's authenticated public encryption key.

Study-note section SEC18

> A sender key pair is needed when the protocol requires its signature or separate recipient role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 8 | lines 61-76

> Asymmetric key cryptography
> • also known as public-key encryption.
> • In public key cryptography,
> o unlike symmetric-key, there are two keys: a private key and a public key.
> ✓ The public key is announced to the public; whereas the private key is kept by the receiver.
> o anyone can send messages using the public key of the receiver for encryption, but only the receiver uses his private key for decryption
> ✓ Secret transmission of key for decryption is not

required
✓ Every entity can generate a key pair and release its public key
✓ Encryption and decryption keys are different but form a unique pair

Computer Security module | PDF page 8 | lines 89-98

• Asymmetric-key cryptosystem Model
o Both users (Bob & Alice) generates a pair of keys
o Each user places one of 2 keys in public register.
✓ This is the public key: each user maintains a collection of public keys obtained from others.
o The companion key is kept private to each.
o If Alice wishes to send a confidential message to Bob, She encrypts the message using Bob's public key.
o When Bob receives the message, he decrypts it using his private key.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 19-1

Computer Science Exit Exam practice collection · Question 183

Question 59 | SM0272

Which one of the following cryptography systems uses the same key for encryption and decryption?

A. Public key cryptography

B. Parallel cryptography

C. Asymmetric cryptography

D. Symmetric cryptography

Hint

Distinguish shared-key architecture, key distribution and bulk-data use.

Answer D: Symmetric cryptography

PDF p.7 defines symmetric encryption as use of the same shared key for both directions; p.8 contrasts the public/private pair.

Why each choice is right or wrong

A - Incorrect: Public-key cryptography uses a key pair. See PDF p.7, PDF p.8.

B - Incorrect: Parallel cryptography is not a same-key class in the module. See PDF p.7, PDF p.8.

C - Incorrect: Asymmetric cryptography uses different paired keys. See PDF p.7, PDF p.8.

D - Correct: Symmetric cryptography explicitly uses the same shared key. See PDF p.7, PDF p.8.

Primary note proof

Study-note section SEC16

Symmetric encryption uses a shared secret and efficiently protects bulk data. Pairwise independent keys for n users require n(n-1)/2 shared keys; a trusted key distribution center changes that architecture. A session key is temporary for a communication context; a master/key-encryption key protects subordinate keys. Distribute keys physically, via an authenticated protected channel, or through a trusted third party. An uncompromised old key can protect a new key, although this does not restore secrecy after the old key is stolen.

Study-note section SEC18

For confidentiality to B, encrypt with B's authenticated public encryption key; B decrypts with B's private key.

Study-note section SEC18

For a signature, A signs with A's private signing key and others verify with A's public key.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 7 | lines 48-54

Symmetric key cryptography
• Uses conventional / secret-key / single-key.
• The secret key shared by both sender and receiver.
• All classical encryption algorithms are private-key.
o Use the same key (shared key) for encryption and decryption process.
• If this key is disclosed communications are compromised.

Computer Security module | PDF page 8 | lines 61-76

Asymmetric key cryptography
• also known as public-key encryption.
• In public key cryptography,
o unlike symmetric-key, there are two keys: a private key and a public key.
✓ The public key is announced to the public; whereas the private key is kept by the receiver.
o anyone can send messages using the public key of the receiver for encryption, but only the receiver uses his private key for decryption
✓ Secret transmission of key for decryption is not required
✓ Every entity can generate a key pair and release its public key

✓ Encryption and decryption keys are different but form a unique pair

Question sources

Computer Science Exit Exam · Sene 2016 · Question 23-3

Computer Science Exit Exam practice collection · Question 185

Question 60 | SM0301

The first step in access control is _____.

A. Audit

B. Authorization

C. Accreditation

D. Authentication

Hint

Separate identity claims from proof, and factor type from factor count.

Answer D: Authentication

The module requires identification and authentication before access and says NAC authenticates users then determines access/actions (PDF pp.4,11). Among these options authentication is the prerequisite, though identification would precede it if offered.

Why each choice is right or wrong

A - Incorrect: Audit records events rather than first proving identity. See PDF p.4, PDF p.11.

B - Incorrect: Authorization determines allowed operations for an established identity. See PDF p.4, PDF p.11.

C - Incorrect: Accreditation is not the identity-checking step defined in the module. See PDF p.4, PDF p.11.

D - Correct: Authentication is the earliest listed access-control step. See PDF p.4, PDF p.11.

Primary note proof

Study-note section SEC07

A common access sequence is identify the subject, authenticate the claim, authorize the operation, then record relevant events. Identification is the username/claim, not its proof. When a sequence begins with authentication, distinguish it from the preceding identification stage. User-to-host means a user proves identity to a host; host-to-user means the host proves its identity to a user; the arrow names the prover and verifier.

Study-note section SEC01

Authentication establishes an identity or other claim; authorization decides permitted actions.

Study-note section SEC01

Accountability links actions to responsible entities through reliable identity and protected logs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Computer Security module | PDF page 11 | lines 88-100

• Network access control (NAC): is an umbrella term for managing access to a net-work. NAC authenticates users logging into the network and determines what data they can access and actions they can perform. NAC also examines the health of the user's computer or mobile device (the endpoints).
• An operating system provides an access enforcement mechanism.
• Two fundamental concepts of access control:
o a protection system that defines the access control specification and
o a reference monitor that is the system's access enforcement mechanism that enforces this specification.

Computer Security module | PDF page 4 | lines 28-35

• Accountability
o Ensuring an entity's action is traceable uniquely to that entity.
o Ascertains the responsibility of an entity (like a person) for its actions and decisions.
o All relevant activities events and operations on a system, e.g., failed and successful authentication attempts, are recorded in a log.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 33-1

Computer Science Exit Exam practice collection · Question 179

Question 61 | SS110

A firewall is used in a system connected to a wide area network to

A. Stop unauthorized access by hackers

B. Scanning for viruses in various files

C. Avoid spreading of fire in the network

D. Fire spreading via network through cables

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer A: Stop unauthorized access by hackers

A firewall filters protected-network access to keep intruders out (PDF p.11). This is a logical security boundary, not a fire-suppression device.

Why each choice is right or wrong

A - Correct: Stopping unauthorized network access is an explicit objective.

B - Incorrect: File virus scanning is not the general firewall function.

C - Incorrect: Physical fire spread is unrelated to packet filtering.

D - Incorrect: Fire through cables is unrelated to the stated network security role.

Primary note proof

Study-note section SEC23

> A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Study-note section SEC11

> A disconnected machine loses many network attack paths yet remains exposed to insiders, removable media, supply-chain attacks and physical access.

Study-note section SEC04

> Ransomware denies access through encryption/locking and extortion.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 22-34

3. Firewall
• a program or network devices (e.g. router) which filters access to a protected network from the internet connection.
• protects a local or network of systems from network-based security threats,
• monitors & controls incoming & outgoing network traffic based on predetermined security rules.
• Firewall Objectives
o Keep intruders, malicious code and unwanted traffic out
o Keep private and sensitive information in
o security wall b/n private (protected) NW & outside word.

Question sources

Computer Science model exam collection · Question 17

Question 62 | SS112

Within OAuth, what component validates the user's identity?

A. "Gre. Authorization Server

B. Resource Server

C. d Oc. Client

D. Browser

Hint

Consider what the attacker can guess or steal, or which component validates a claim.

Answer A: Authorization server

SEC08 assigns resource-owner authentication/approval to the authorization server, which may delegate identity checks. The resource server checks access tokens for protected resources; the client seeks limited access and the browser is a user agent. OAuth is authorization rather than a complete login identity protocol, but A is the matching component role under the described flow.

Why each choice is right or wrong

A - Correct: The authorization server authenticates the resource owner or delegates that check and obtains approval.

B - Incorrect: The resource server validates access tokens for protected data rather than being the owner-authentication component.

C - Incorrect: The client requests limited access; it is not the trusted identity-validation role defined here.

D - Incorrect: The browser carries user interaction and redirects but is not the authorization server's authentication authority.

Primary note proof

Study-note section SEC08

OAuth is an authorization framework: a client obtains limited access tokens, the authorization server authenticates/obtains approval from the resource owner, and the resource server validates tokens for protected data. The authorization server can delegate identity checks. OAuth alone does not define a complete end-user login identity protocol; OpenID Connect adds an identity layer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Question sources

Computer Science model exam collection · Question 19

Question 63 | SS136

Which protocol is specifically designed for secure remote login and remote command execution between a local client and a remote computer over an untrusted network?

A. Authentication, Authorization and Accounting (AAA)

B. Simple Network Management Protocol (SNMP)

C. Secure Shell (SSH)

D. Secure socket layer (SSL)

Hint

Match the communication task to the protocol and trace key handling separately from signing.

Answer C: Secure Shell (SSH)

SSH provides secure remote login, command execution and related services over an untrusted network. SSL and its successor TLS provide protected channels but do not by themselves define the remote-login and command-execution service requested here. AAA names authentication, authorization and accounting functions rather than a remote-shell protocol. SNMP supplies network-management functions. Specifying the service, rather than merely a secure local-to-remote channel, makes SSH the unique answer among these choices.

Why each choice is right or wrong

A - Incorrect: AAA describes authentication, authorization and accounting functions, not a remote-login protocol.

B - Incorrect: SNMP provides device-management functions rather than the requested remote-login and command-execution service.

C - Correct: Correct: SSH directly provides secure remote login and remote command execution.

D - Incorrect: SSL/TLS protects communication channels but does not itself specify the requested remote-shell service.

Primary note proof

Study-note section SEC22

> SSH protects remote login, commands and file transfer such as SFTP; it supports host keys and several client-authentication methods.

Study-note section SEC-S03

> AAA names authentication, authorization and accounting functions, rather than a particular remote-shell channel. SSH expands to Secure Shell and provides secure remote login and related network services. PGP expands to Pretty Good Privacy; its message-protection design combines conventional and public-key encryption with digital signatures. Protocol purpose matters when several technologies all contribute to security.

Study-note section SEC21

> TLS succeeds obsolete SSL and supplies an authenticated protected channel.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 2-7

> o A cryptographic protocol for operating NW services
> securely over an unsecured NW.
> o Best-known application: for remote login to computer
> systems by users.

o Common applications: remote command-line login and execution.

Question sources

Computer Science model exam collection · Question 43

Question 64 | SS150

Which of the following refers to unauthorized disclosure of information?

A. Integrity

B. Authorization

C. Confidentiality

D. Authentication

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer C: Confidentiality is the property violated by unauthorized disclosure

SEC01 defines confidentiality as restricting disclosure, so unauthorized disclosure is a breach of that property. Integrity concerns improper changes, authorization determines permitted operations and authentication establishes claims. The answer names the affected security goal rather than calling unauthorized disclosure itself a protective mechanism.

Why each choice is right or wrong

A - Incorrect: Integrity restricts improper changes, not disclosure of unchanged information.

B - Incorrect: Authorization decides which actions are permitted rather than naming the secrecy property harmed.

C - Correct: Confidentiality restricts disclosure and is directly violated when information is revealed without authorization.

D - Incorrect: Authentication establishes identity/claims rather than defining protection against disclosure.

Primary note proof

Study-note section SEC01

Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC01

Authentication establishes an identity or other claim; authorization decides permitted actions.

Study-note section SEC01

Accountability links actions to responsible entities through reliable identity and protected logs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 71-78

• Confidentiality: Data confidentiality and Privacy
o is the prevention of unauthorized disclosure of information.
o In other words, confidentiality means keeping information private or safe.
o it may be important for military, business or personal reasons.
o it may also be known as privacy or secrecy.

Computer Security module | PDF page 3 | lines 79-86

• Integrity: covers Data integrity and System integrity
o the prevention of unauthorized writing or modification of information.
o Integrity in a computer system means that there is an external consistency in the system - everything is as it is expected to be.
o Data integrity: the data stored on the computer is the same as what is intended.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Question sources

Computer Science model exam collection · Question 57

Question 65 | SS152

Which benefit follows directly from network ACL match-and-permit/deny rules, independently of optional traffic counters or malware-inspection features?

A. Virus detection.

B. ACLs classify and organize network traffic.

C. ACLs provide high network availability.

D. ACLs monitor the number of bytes and packets.

Hint

Identify subject, object and permitted operation before comparing policy models.

Answer B: ACLs classify and organize network traffic.

A network ACL matches traffic attributes such as addresses, protocols and ports and applies permission rules. Distinguishing the matching packet groups directly supports the offered classification and organization benefit. Malware inspection is a separate detection function, and a permission list does not guarantee high availability. An implementation may attach counters to rules, but those optional measurement features are outside the basic ACL permission function specified here. B is the direct consequence of that function.

Why each choice is right or wrong

A - Incorrect: Incorrect: matching permission rules is not malware-content detection.

B - Correct: Correct: matching packet conditions distinguishes traffic groups for permit/deny treatment.

C - Incorrect: Incorrect: a permission list does not by itself guarantee continued service availability.

D - Incorrect: Incorrect: byte/packet counters are optional measurement features, excluded from the basic permission-rule function asked about.

Primary note proof

Study-note section SEC09

> Subjects request operations on objects. An ACL associates an object/resource with allowed subjects or actions; a network ACL matches traffic such as address, protocol and port. Classification and filtering are direct effects; high availability or malware detection is not guaranteed by an ACL.

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 80-87

6. Access control
• Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Computer Security module | PDF page 11 | lines 35-56

• Categorizes of firewalls
a) Network firewalls: Filter traffic b/n two or more networks and run-on NW hardware.
✓ Implemented at a specific point in the network path and protects all computers on the internal side of the firewall from all computers on the external side of the fire wall.
b) Host-based firewalls: Run on host computers and control network traffic in/out of those machines.
a) Packet Filtering Router: Examines TCP/IP header information of network data packets going in both directions. Applies set of rules to each incoming IP packets & then forwards or discards the packets.
b) Application-Level Gateway (application firewall / proxy server): runs special software that acts as a proxy for a service request. Proxy server receives requests for Web pages, accesses the Web server on behalf of the external client, and returns the requested pages to the users.
c)Circuit Level Gateway: Standalone Software that sets up two TCP connections (inside & outside). It determines which connections will be allowed.

Question sources

Computer Science model exam collection · Question 59

Question 66 | SS185

Which of the following is not an approach used by IT security specialists to enhance the security level of the network?

A. Use of Intrusion detection system

B. Use of Intrusion prevention system

C. Use of Wana cry protocol

D. Use of physical security

Hint

Distinguish a malicious-software name from a defensive control's function.

Answer C: “Wana cry protocol” is not a defensive security approach

IDS, IPS and physical controls are legitimate security approaches in SEC24 and SEC11. The option called “Wana cry protocol” supplies no defensive mechanism; treating a ransomware-like name as a security protocol is unsupported. SEC04 defines ransomware as denial/extortion, the opposite of a protective approach. Therefore C is the exception rather than a new protocol endorsed here.

Why each choice is right or wrong

A - Incorrect: An IDS observes and alerts on suspicious activity, a legitimate detective security approach.

B - Incorrect: An IPS can enforce blocking and is a legitimate preventive security approach.

C - Correct: The offered name establishes no security protocol or protective mechanism; ransomware is harmful rather than defensive.

D - Incorrect: Physical protection against unauthorized access and environmental hazards is an explicit security-control category.

Primary note proof

Study-note section SEC-S02

> WannaCry, also known as WannaCrypt, is ransomware with worm-like propagation. It encrypts files and demands payment to restore access. Its name identifies malicious software rather than a network-security protocol. Malware names must be distinguished from defensive controls such as intrusion detection, intrusion prevention and physical protection.

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline.

Study-note section SEC11

> Protect equipment against theft, unauthorized connections, heat, humidity, water, electrical surges and fire.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 89-99

> o Encryption: primary control that protects CIA, by
> which Cleartext scambled into ciphertext.
> o Software controls: password checker, virus scanner,
> IDS (intrusion detection system)
> o Hardware controls: provide higher degree of security;
> Locks, Smart cards, dongles, hadware keys...

o Policies and procedures: Alignment with users' legal and ethical standards; eg. password policy
o Physical controls: natural and man-made disaster protection; walls, locks, guards, security, cameras, backup copies and archives

Computer Security module | PDF page 11 | lines 65-68

4. Physical security
• Without physical security to protect hardware (i.e., doors that lock) nothing else about a computer system can be called secure

Computer Security module | PDF page 5 | lines 21-35

• Malware Attack: A generic term for software that has malicious purpose.
o Viruses
o Worms
o Trojan horses
o Spy-wares
o Zombie
o Botnets
o Bacterium
o Logic bomb
o Backdoor(Trapdoor)
o Adware
o Flooders (DoS client)
o Key loggers
o Rootkit

Question sources

Computer Science model exam collection · Question 92

Question 67 | AM0524

What is the primary goal of computer security?

A. Confidentiality

B. Integrity

C. Availability

D. All of the above

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer D: All of the above

The module explicitly defines computer security around all three objectives confidentiality, integrity and availability (PDF p.2). Choosing only one omits other primary goals.

Why each choice is right or wrong

A - Incorrect: Confidentiality is included but incomplete alone. See PDF p.2.

B - Incorrect: Integrity is included but incomplete alone. See PDF p.2.

C - Incorrect: Availability is included but incomplete alone. See PDF p.2.

D - Correct: All three comprise the stated CIA objective set. See PDF p.2.

Primary note proof

Study-note section SEC01

> Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC01

> Authentication establishes an identity or other claim; authorization decides permitted actions.

Study-note section SEC01

> Data security protects information at rest, in use and in transit, including accidental corruption as well as intentional attacks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 2 | lines 84-90

> • Computer Security: (W. Stallings)
> o Protection to an automated information system in order to attain the applicable objectives of preserving the confidentiality, integrity, and availability (CIA) of information system resources (includes hardware, software, firmware, information/data, and telecommunications).

Question sources

Computer Science MCQ practice collection · Question 524

Question 68 | AM0525

What is the term used for a software program that attempts to gain unauthorized access to computer systems?

A. Firewall

B. Malware

C. Encryption

D. VPN

Hint

Separate propagation, disguise and payload when classifying malicious software.

Answer B: Malware

PDF p.5 calls software with malicious purpose malware and describes intrusion as attempted unauthorized access. A malicious program seeking unauthorized access therefore fits the broad malware class.

Why each choice is right or wrong

A - Incorrect: Firewall is a protective filtering mechanism, not the malicious program. See PDF p.5.

B - Correct: Malware is the umbrella for software with malicious purpose. See PDF p.5.

C - Incorrect: Encryption is a transformation for protected communication, not a malicious program class. See PDF p.5.

D - Incorrect: VPN is named as an application, not the malicious-software category. See PDF p.5.

Primary note proof

Study-note section SEC04

> A virus replicates by attaching to a host program or document; execution of the infected host spreads it. A worm replicates independently, commonly across networks. A Trojan disguises malicious behavior as something useful and does not require self-replication. These labels describe different properties, so a malware package may contain several components.

Study-note section SEC04

> Malware is the umbrella term, not a specific replication method.

Study-note section SEC04

> Ransomware denies access through encryption/locking and extortion.

Related reading (context, not primary note proof)

Study-note section SEC23

Study-note section SEC12

Study-note section SEC22

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 21-35

• Malware Attack: A generic term for software that has malicious purpose.
o Viruses
o Worms
o Trojan horses
o Spy-wares
o Zombie
o Botnets
o Bacterium
o Logic bomb
o Backdoor(Trapdoor)
o Adware
o Flooders (DoS client)
o Key loggers
o Rootkit

Computer Security module | PDF page 5 | lines 37-43

• Hacking /Intrusion/ Attack: any attempt to intrude or gain unauthorized access to your system either via some operating system flaw or other means. The purpose may or may not be for malicious purposes.
• Cracking: is hacking conducted for malicious purposes.
• Denial of Service (DoS) Attack: Blocking access from legitimate users

Question sources

Computer Science MCQ practice collection · Question 525

Question 69 | AM0526

What is the practice of hiding confidential data within other data called?

A. Encryption

B. Decryption

C. Steganography

D. Intrusion detection

Hint

Trace message transformations and separate data, keys, algorithms and codebreaking.

Answer C: Steganography

Hiding confidential data inside another data object aims to conceal the message's existence. SEC12 calls that steganography and distinguishes it from encryption, which conceals meaning while leaving ciphertext observable. Decryption reverses encryption, and intrusion detection observes suspicious activity rather than embedding a hidden message.

Why each choice is right or wrong

A - Incorrect: Encryption conceals message meaning rather than specifically hiding its existence within another carrier.

B - Incorrect: Decryption recovers plaintext from ciphertext; it does not embed a hidden message.

C - Correct: Steganography conceals a message's existence by hiding it within other data.

D - Incorrect: Intrusion detection observes and alerts about suspicious activity instead of hiding confidential content.

Primary note proof

Study-note section SEC12

> Plaintext is the original message, ciphertext its encrypted representation, a key is input to an algorithm, encryption computes C=E_k(P), and decryption recovers P=D_k(C). The five conventional symmetric ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. Cryptanalysis seeks recovery or other breaks without the authorized secret; cryptology includes cryptography and cryptanalysis. Steganography hides that a message exists; encryption hides its meaning.

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline.

Study-note section SEC24

> Signature detection recognizes known patterns; anomaly detection compares with a baseline and may detect novel behavior.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 2-12

> o Encipher (encrypt) - the process of converting
> plaintext to ciphertext
> o Decipher (decrypt) - recovering plaintext from
> ciphertext
> o Cryptography - the study of encryption
> principles/methods
> o Cryptanalysis (codebreaking) - the study of
> principles/ methods of breaking ciphertext without
> knowing key

o Cryptology - the field of study which deals with both cryptography and cryptanalysis

Question sources

Computer Science MCQ practice collection · Question 526

Question 70 | AM0527

Which of the following authentication methods verifies a user's identity based on a unique physical characteristic?

A. Password authentication

B. Biometric authentication

C. Token authentication

D. Two-factor authentication

Hint

Separate identity claims from proof, and factor type from factor count.

Answer B: Biometric authentication

The module identifies face, retina, iris and fingerprint as biometrics (PDF p.11), and gives retina scanning as identity proof (p.4). A unique physical characteristic therefore selects biometrics.

Why each choice is right or wrong

A - Incorrect: Password authentication uses supplied secret knowledge, not a physical body trait. See PDF p.4, PDF p.11.

B - Correct: Biometric authentication uses the listed bodily examples. See PDF p.4, PDF p.11.

C - Incorrect: Physical tokens are cards, distinguished from biometrics. See PDF p.4, PDF p.11.

D - Incorrect: Two-factor counts combined factors and does not name the body-trait method. See PDF p.4, PDF p.11.

Primary note proof

Study-note section SEC07

Knowledge is a password/PIN, possession is a token/card, and inherence is a biometric such as a fingerprint. Behavioral biometrics include typing rhythm or handwriting; some textbooks call these “something one does.” A body characteristic is not “three-factor authentication”: one biometric is one factor. Password plus PIN remains one factor class; card plus PIN is two. An OTP’s factor depends on its delivery/generation mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 74-79

• Most OS authentication is handled through user names and passwords. Biometric (e.g. voice, face, retina, iris, fingerprint) and physical token-based (swipe cards, pin-generating cards) authentication are sometimes used to augment simple passwords, but the costs and accuracy of the technology limit their adoption.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Question sources

Computer Science MCQ practice collection · Question 527

Question 71 | AM0528

What is the process of converting plaintext into ciphertext called?

A. Encryption

B. Decryption

C. Hashing

D. Key exchange

Hint

Trace message transformations and separate data, keys, algorithms and codebreaking.

Answer A: Encryption

PDF p.6 directly defines encryption/enciphering as conversion of plaintext to ciphertext. Hashing instead produces a fixed-size digest (p.10).

Why each choice is right or wrong

A - Correct: Encryption exactly names the forward conversion. See PDF p.6, PDF p.10.

B - Incorrect: Decryption is the reverse conversion. See PDF p.6, PDF p.10.

C - Incorrect: Hashing produces a digest rather than the defined ciphertext representation. See PDF p.6, PDF p.10.

D - Incorrect: Key exchange distributes/agrees keys rather than converting this message. See PDF p.6, PDF p.10.

Primary note proof

Study-note section SEC12

Plaintext is the original message, ciphertext its encrypted representation, a key is input to an algorithm, encryption computes C=E_k(P), and decryption recovers P=D_k(C). The five conventional symmetric ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. Cryptanalysis seeks recovery or other breaks without the authorized secret; cryptology includes cryptography and cryptanalysis. Steganography hides that a message exists; encryption hides its meaning.

Study-note section SEC20

A cryptographic hash maps arbitrary-length input to a fixed-length digest.

Study-note section SEC18

Authenticate the exchange and derive separate traffic keys.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 2-12

o Encipher (encrypt) - the process of converting
plaintext to ciphertext
o Decipher (decrypt) - recovering plaintext from
ciphertext
o Cryptography - the study of encryption
principles/methods
o Cryptanalysis (codebreaking) - the study of
principles/ methods of breaking ciphertext without
knowing key
o Cryptology - the field of study which deals with both
cryptography and cryptanalysis

Computer Security module | PDF page 10 | lines 23-30

o A hash function H takes a variable-size message m as
input and produces a fixed-size output, referred to as a
hash code or message digest or hash value H (m)
o no key is used in this algorithm.
o A fixed-length hash value is computed as per the
plain text
✓ that makes it impossible for the contents of the plain
text to be recovered.

Question sources

Computer Science MCQ practice collection · Question 528

Question 72 | AM0529

Which vulnerability writes beyond an allocated buffer and may thereby allow arbitrary code execution on the target system?

A. Phishing

B. Buffer overflow

C. SQL injection

D. Cross-site scripting

Hint

Identify the unsafe operation or input boundary rather than only a possible consequence.

Answer B: Buffer overflow

The decisive mechanism is an out-of-bounds write to a buffer. That is a buffer overflow, which can corrupt data or control information and may lead to code execution. Arbitrary code execution alone would not uniquely identify it. Phishing relies on deceptive interaction, SQL injection changes query syntax and XSS executes attacker script in a victim browser. The added memory-write condition distinguishes the mechanisms without claiming every overflow executes code.

Why each choice is right or wrong

A - Incorrect: Phishing deceives a user rather than defining an out-of-bounds memory write.

B - Correct: A buffer overflow is the specified write beyond allocated bounds.

C - Incorrect: SQL injection changes SQL syntax through unsafe input construction.

D - Incorrect: XSS concerns attacker script execution in a victim browser's site context.

Primary note proof

Study-note section SEC25

> A buffer overflow writes beyond allocated bounds and can corrupt data or control information; code execution is a possible consequence, not its definition or an inevitable outcome. An array of eight bytes cannot store eight text characters plus a terminating zero. Bounds-check lengths and indexes, consider integer overflow, use safe abstractions and validate library return values. Canaries, non-executable memory and address randomization reduce exploitability but do not repair the underlying write.

Study-note section SEC26

> XSS causes attacker-controlled script to execute in a victim's browser under the site's origin.

Study-note section SEC26

SQL injection occurs when untrusted values alter SQL syntax through unsafe string construction.

Related reading (context, not primary note proof)

Study-note section SEC05

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Computer Security module | PDF page 10 | lines 71-73

o Database Based :
✓ identify the security exposure in the database systems to prevent from SQL Injections.

Computer Security module | PDF page 11 | lines 12-16

• Secure Web servers:
o The main vulnerability of web applications is Cross-Site Scripting (XSS).
o Securing the website or web application itself and the network around it.

Question sources

Computer Science MCQ practice collection · Question 529

Question 73 | AM0530

Which of the following is an example of a social engineering attack?

A. Denial-of-Service (DoS) attack

B. Distributed Denial-of-Service (DDoS) attack

C. Phishing attack

D. Brute-force attack

Hint

Identify the trust claim or communication mapping being abused.

Answer C: Phishing

SEC05 defines phishing through deception: impersonating a trusted source to obtain secrets or prompt unsafe actions. That is social engineering. DoS/DDoS describe service disruption and brute force describes repeated credential guesses; neither requires the deceptive human interaction that distinguishes the requested example.

Why each choice is right or wrong

A - Incorrect: DoS exhausts or disables a service; that mechanism is not inherently social deception.

B - Incorrect: DDoS distributes service-disruption traffic and does not specifically describe manipulating a person.

C - Correct: Phishing deceives a user through trusted-source impersonation to induce disclosure or unsafe action.

D - Incorrect: Brute force enumerates credential guesses rather than persuading a person through impersonation.

Primary note proof

Study-note section SEC05

> Phishing impersonates a trusted source to obtain secrets or induce an unsafe action; spear phishing targets a specific person. Verify a request through an independently known channel. Shoulder surfing observes entry; pretexting invents a reason for disclosure. Monitoring software used with authorization is not necessarily an attack merely because it records activity.

Study-note section SEC08

> Brute force enumerates guesses; dictionary attacks prioritize likely words; credential stuffing tries previously leaked credentials on another service.

Study-note section SEC06

> DoS exhausts or disables a resource; distributed DoS uses multiple sources, often bots or reflected/amplified responses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 65-74

> • Reconnaissance: The hackers first identifies a target to
> launch an attack, extract maximum information regarding
> this target, understand its vulnerabilities, and then only
> explore the best ways to exploit it.
> o Examples: Credential Theft using Keyloggers and
> Spyware; Identity Theft using spam e-mail, phishing
> attack and social engineering attack
> • Access: Ones the attacker Reconnaissance the target, S/he

starts a successful exploitation and continued access to the system. Then, the attacker is in control of the target.

Computer Security module | PDF page 5 | lines 50-60

• Password attacks: guessing password; social engineering attack, Dictionary attack
• Cryptographic Attack:
A. Frequency Analysis attack: Guess values based on frequency of occurrence
B. Brute-force attack: The attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried to achieve success.
C. Cryptanalysis: the process of breaking an encrypted code.

Question sources

Computer Science MCQ practice collection · Question 530

Question 74 | AM0531

What is the purpose of a firewall in computer security?

A. To encrypt network traffic

B. To monitor and control network traffic

C. To detect and remove malware

D. To authenticate users

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer B: To monitor and control network traffic

PDF p.11 explicitly assigns monitoring and controlling incoming/outgoing traffic under security rules to firewalls. Encryption and identity proof are different operations.

Why each choice is right or wrong

A - Incorrect: Traffic encryption is not the firewall definition. See PDF p.4, PDF p.11.

B - Correct: Monitoring and controlling traffic is the exact stated role. See PDF p.4, PDF p.11.

C - Incorrect: Keeping malicious code out is an objective, not a general ability to detect and remove installed malware. See PDF p.4, PDF p.11.

D - Incorrect: User authentication is a separate identity-proof operation. See PDF p.4, PDF p.11.

Primary note proof

Study-note section SEC23

A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Study-note section SEC07

User-to-host means a user proves identity to a host; host-to-user means the host proves its identity to a user; the arrow names the prover and verifier.

Study-note section SEC07

When a sequence begins with authentication, distinguish it from the preceding identification stage.

Related reading (context, not primary note proof)

Study-note section SEC12

Study-note section SEC04

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 22-34

3. Firewall
• a program or network devices (e.g. router) which filters access to a protected network from the internet connection.
• protects a local or network of systems from network-based security threats,
• monitors & controls incoming & outgoing network traffic based on predetermined security rules.
• Firewall Objectives
o Keep intruders, malicious code and unwanted traffic out
o Keep private and sensitive information in
o security wall b/n private (protected) NW & outside word.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls

o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Question sources

Computer Science MCQ practice collection · Question 531

Question 75 | AM0532

A login already uses a password. Which listed input can add an independent possession factor when generated by a separate token controlled by the user?

A. Username

B. Password

C. Security question

D. One-time password (OTP)

Hint

Separate identity claims from proof, and factor type from factor count.

Answer D: One-time password (OTP)

A password supplies knowledge. A one-time password generated by a separate user-controlled token can supply possession, making the combination two-factor under the stated mechanism. Another password or secret security-question answer remains knowledge rather than adding an independent class. A username identifies the subject but is not proof. Specifying how the OTP is generated matters because an OTP's factor class does not follow from the word one-time alone.

Why each choice is right or wrong

A - Incorrect: A username identifies the subject and is not possession proof.

B - Incorrect: Another password remains a knowledge factor.

C - Incorrect: A secret answer is another knowledge input, not an independent possession class.

D - Correct: A separately controlled token generating the OTP supplies possession under the assumption.

Primary note proof

Study-note section SEC07

Knowledge is a password/PIN, possession is a token/card, and inherence is a biometric such as a fingerprint. Behavioral biometrics include typing rhythm or handwriting; some textbooks call

these “something one does.” A body characteristic is not “three-factor authentication”: one biometric is one factor. Password plus PIN remains one factor class; card plus PIN is two. An OTP’s factor depends on its delivery/generation mechanism.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 74-79

• Most OS authentication is handled through user names and passwords. Biometric (e.g. voice, face, retina, iris, fingerprint) and physical token-based (swipe cards, pin-generating cards) authentication are sometimes used to augment simple passwords, but the costs and accuracy of the technology limit their adoption.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Question sources

Computer Science MCQ practice collection · Question 532

Question 76 | AM0533

What is the term for a malicious activity where an attacker tries all possible combinations of passwords to gain unauthorized access?

A. Brute-force attack

B. Denial-of-Service (DoS) attack

C. SQL injection

D. Man-in-the-middle (MITM) attack

Hint

Consider what the attacker can guess or steal, or which component validates a claim.

Answer A: Brute-force attack

The module defines brute force as trying every possible key and separately names password guessing (PDF p.5). Applying that exhaustive-search method to all password combinations yields a brute-force password attack.

Why each choice is right or wrong

A - Correct: Exhaustive possible-combination testing is the brute-force method. See PDF p.5.

B - Incorrect: DoS blocks access rather than exhaustively guessing a secret. See PDF p.5.

C - Incorrect: SQL injection is not an exhaustive-password-search method in the module. See PDF p.5.

D - Incorrect: A man-in-the-middle is not the stated exhaustive-search process. See PDF p.5.

Primary note proof

Study-note section SEC08

> Brute force enumerates guesses; dictionary attacks prioritize likely words; credential stuffing tries previously leaked credentials on another service. Online attempts face server rate limits; offline attacks against stolen verifiers do not. Short passwords can fail in either setting. Use unique long secrets and password managers, rate limiting and suitable MFA; store salted slow password hashes, not reversible plaintext passwords. A salt prevents shared precomputed lookup, but is not secret.

Study-note section SEC05

> Monitoring software used with authorization is not necessarily an attack merely because it records activity.

Study-note section SEC26

> SQL injection occurs when untrusted values alter SQL syntax through unsafe string construction.

Related reading (context, not primary note proof)

Study-note section SEC06

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 50-60

> • Password attacks: guessing password; social engineering attack, Dictionary attack
> • Cryptographic Attack:
> A. Frequency Analysis attack: Guess values based on frequency of occurrence
> B. Brute-force attack: The attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried to achieve success.

C. Cryptanalysis: the process of breaking an encrypted code.

Question sources

Computer Science MCQ practice collection · Question 533

Question 77 | AM0534

Which statement defines a buffer overflow by its underlying memory-write mechanism, rather than merely naming a possible consequence?

A. A vulnerability that allows unauthorized access to a system's memory

B. A vulnerability that allows attackers to intercept network traffic

C. A vulnerability in which a write exceeds an allocated buffer’s bounds

D. A vulnerability that allows unauthorized access to a system's files

Hint

Identify the unsafe operation or input boundary rather than only a possible consequence.

Answer C: A vulnerability in which a write exceeds an allocated buffer’s bounds

A buffer overflow is defined by writing beyond an allocated buffer’s bounds. That unsafe write can corrupt data or control information, and code execution is possible but not inevitable. Generic unauthorized memory or file access describes broader outcomes, while interception of network traffic describes a different kind of activity. The choice identifies the mechanism; the question asks for a definition rather than a consequence.

Why each choice is right or wrong

A - Incorrect: Generic unauthorized memory access is broader than the specific out-of-bounds write.

B - Incorrect: Network interception is not the defining write mechanism.

C - Correct: This states the defining write beyond allocated buffer bounds.

D - Incorrect: Unauthorized file access is a possible harm rather than the definition.

Primary note proof

Study-note section SEC25

A buffer overflow writes beyond allocated bounds and can corrupt data or control information; code execution is a possible consequence, not its definition or an inevitable outcome. An array of eight bytes cannot store eight text characters plus a terminating zero. Bounds-check lengths and indexes, consider integer overflow, use safe abstractions and validate library return values. Canaries, non-executable memory and address randomization reduce exploitability but do not repair the underlying write.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 53-59

• Example: If a company has antivirus software but does not use an anti-virus and keep the virus signatures up-to-date, this is vulnerability. The company is vulnerable to virus attacks. Similarly, if you fail to routinely update your operating systems or application software, these will remain vulnerable to software problems ("bugs") that have been identified and patched

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Computer Science MCQ practice collection · Question 534

Question 78 | AM0535

Which statement defines SQL injection by its mechanism rather than merely describing a possible security consequence?

A. A vulnerability that allows attackers to intercept network traffic

B. A vulnerability that allows attackers to execute arbitrary code

C. A vulnerability that allows unauthorized access to a system's memory

D. Untrusted input changes SQL syntax through unsafe query construction

Hint

Locate where untrusted data can become instructions and where they execute.

Answer D: Untrusted input changes SQL syntax through unsafe query construction

SQL injection occurs when untrusted values become part of SQL syntax through unsafe query construction. That can change query behavior or expose protected records, but unauthorized database access alone does not define the flaw. Network interception and memory access are other broad harms, and arbitrary code execution can occur through multiple mechanisms. The choice identifies the query-syntax boundary that parameter binding is intended to protect.

Why each choice is right or wrong

A - Incorrect: Network interception does not define unsafe SQL construction.

B - Incorrect: Code execution is a broad possible consequence rather than the SQL-specific mechanism.

C - Incorrect: Unauthorized memory access does not identify the query-syntax boundary.

D - Correct: This states how untrusted input changes SQL command structure.

Primary note proof

Study-note section SEC26

> SQL injection occurs when untrusted values alter SQL syntax through unsafe string construction. A tautology can change which rows match. Bind data values in prepared statements, allow-list identifiers that cannot be parameters, and restrict the application's database account. Stored procedures/ORMs are safe only when they avoid unsafe dynamic SQL.

Study-note section SEC26

> Use safe text insertion and context-specific encoding; carefully sanitize allowed HTML.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 71-73

> o Database Based :
> ✓ identify the security exposure in the database systems to prevent from SQL Injections.

Computer Security module | PDF page 10 | lines 56-62

> ✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
> ✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Computer Science MCQ practice collection · Question 535

Question 79 | AM0536

Which statement defines cross-site scripting by its execution context, rather than merely naming one possible effect?

A. A vulnerability that allows unauthorized access to a system's memory

B. Attacker-controlled script executes in a victim’s browser under the vulnerable site’s origin

C. A vulnerability that allows attackers to manipulate web page content

D. A vulnerability that allows unauthorized access to a database

Hint

Locate where untrusted data can become instructions and where they execute.

Answer B: Attacker-controlled script executes in a victim’s browser under the vulnerable site’s origin

XSS causes attacker-controlled script to run in a victim’s browser under the vulnerable site’s origin. Page-content manipulation can be an effect of that execution but is not a complete definition, since other flaws can also change content. Generic memory or database access likewise does not identify the browser-origin mechanism. The answer states the relevant execution context without treating all arbitrary code execution as XSS.

Why each choice is right or wrong

A - Incorrect: Unauthorized memory access does not define the browser-origin script mechanism.

B - Correct: This states the script execution and origin context that define XSS.

C - Incorrect: Page-content manipulation is a possible effect rather than the defining execution context.

D - Incorrect: Unauthorized database access is not the definition of XSS.

Primary note proof

Study-note section SEC26

> XSS causes attacker-controlled script to execute in a victim’s browser under the site’s origin. Stored content persists; reflected content returns in a response; DOM-based XSS can arise from unsafe browser sinks. Use safe text insertion and context-specific encoding; carefully sanitize allowed HTML. CSP is added protection, not the sole fix.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 12-16

> • Secure Web servers:
> o The main vulnerability of web applications is Cross-Site Scripting (XSS).
> o Securing the website or web application itself and the network around it.

Question sources

Computer Science MCQ practice collection · Question 536

Question 80 | AM0537

What is the purpose of input validation in program security?

A. To ensure the confidentiality of data

B. To prevent unauthorized access to a system

C. To detect and block malicious input

D. To encrypt sensitive data during transmission

Hint

Identify the unsafe operation or input boundary rather than only a possible consequence.

Answer C: Validate and reject unacceptable input

Input validation checks whether data meets expected type, length, range and structure, allowing invalid or malicious inputs to be rejected. SEC25 requires server-side checks because clients are controllable. This supports C without claiming validation alone stops every attack: parameterized queries and context-specific output encoding remain separate protections.

Why each choice is right or wrong

A - Incorrect: Confidentiality is not the specific purpose of checking input type, range, length and structure.

B - Incorrect: Authentication/authorization enforce access; input validation instead checks whether supplied values are acceptable.

C - Correct: Validation detects values outside expected constraints so the program can reject unsafe or invalid input.

D - Incorrect: Encrypting transmission is channel protection rather than input validation.

Primary note proof

Study-note section SEC25

> Validate expected type, length, range and structure at trust boundaries. Server validation is essential because clients are controllable. Encode output for its destination; validation is not a universal substitute for query parameterization or HTML-context encoding. Security patches repair known weaknesses; a zero-day may lack a vendor fix, but segmentation and other mitigations can still help.

Study-note section SEC09

> Allowing SELECT but denying UPDATE is an access-control decision that helps integrity; it does not by itself enforce all valid business-state transitions.

Study-note section SEC09

Deny by default, mediate every access and revoke stale permissions.

Related reading (context, not primary note proof)

Study-note section SEC07

Study-note section SEC21

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Computer Security module | PDF page 10 | lines 71-73

o Database Based :
✓ identify the security exposure in the database systems to prevent from SQL Injections.

Question sources

Computer Science MCQ practice collection · Question 537

Question 81 | AM0538

What is the role of secure coding practices in program security?

A. To prevent code duplication and improve program efficiency

B. To ensure compliance with industry standards and regulations

C. To minimize the number of bugs and vulnerabilities in code

D. To facilitate code collaboration among developers

Hint

Trace timing over shared state or distinguish finding a weakness from demonstrating consequences.

Answer C: Reduce software defects and vulnerabilities

Secure development aims to reduce exploitable defects through review, safe input/memory handling, least privilege and related practices described in SEC25-27. C states that direct aim. Efficiency, compliance and collaboration may be benefits of broader engineering practices, but none substitutes for minimizing flaws that can violate the program's security requirements.

Why each choice is right or wrong

A - Incorrect: Avoiding duplication and improving efficiency are general engineering concerns rather than the specific security aim.

B - Incorrect: Compliance can be relevant but does not directly state the defect-reduction purpose of secure coding.

C - Correct: Secure coding reduces bugs and exploitable vulnerabilities through safer design and implementation.

D - Incorrect: Collaboration helps engineering but is not itself the security objective of minimizing exploitable code flaws.

Primary note proof

Study-note section SEC27

> A race occurs when correctness depends on timing/interleaving over shared state. In a check-then-open file operation, another actor may replace the file between the check and use. In a balance update, two workers can both read100 and each write90, losing one debit. Protect the whole invariant with suitable atomic operations, locks or database transactions; locking an unrelated variable does not help.

Study-note section SEC27

> Build threat modeling, review, least privilege, dependency management and safe error handling into development.

Study-note section SEC25

> Canaries, non-executable memory and address randomization reduce exploitability but do not repair the underlying write.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 53-59

> • Example: If a company has antivirus software but does not use an anti-virus and keep the virus signatures up-to-date, this is vulnerability. The company is vulnerable to virus attacks. Similarly, if you fail to routinely update your operating systems or application software, these will remain vulnerable to software problems ("bugs") that have been identified and patched

Computer Security module | PDF page 10 | lines 56-62

> ✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
> ✓ Penetration Testing: an authorized simulated

attack on a computer system, performed to evaluate
the security of the system.

Question sources

Computer Science MCQ practice collection · Question 538

Question 82 | AM0539

What is the purpose of access control mechanisms in program security?

A. To prevent unauthorized access to sensitive resources

B. To monitor and log system activities for auditing purposes

C. To encrypt data during storage and transmission

D. To protect against malware and viruses

Hint

Identify subject, object and permitted operation before comparing policy models.

Answer A: To prevent unauthorized access to sensitive resources

Access enforcement authorizes subjects to perform operations on objects (PDF p.11), implementing limitation to authorized users (p.4). That protects sensitive resources from unauthorized access.

Why each choice is right or wrong

A - Correct: Restricting resource operations to authorized subjects is exactly access control. See PDF p.4, PDF p.11.

B - Incorrect: Logging is the separate accountability function. See PDF p.4, PDF p.11.

C - Incorrect: Encryption transforms data rather than assigning object permissions. See PDF p.4, PDF p.11.

D - Incorrect: Malware defense is broader and does not define access control. See PDF p.4, PDF p.11.

Primary note proof

Study-note section SEC09

Subjects request operations on objects. An ACL associates an object/resource with allowed subjects or actions; a network ACL matches traffic such as address, protocol and port. Classification and filtering are direct effects; high availability or malware detection is not guaranteed by an ACL.

Study-note section SEC01

Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it.

Study-note section SEC09

RBAC gives permissions to roles and assigns users to roles.

Related reading (context, not primary note proof)

Study-note section SEC12

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Computer Security module | PDF page 11 | lines 80-87

6. Access control
• Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Computer Security module | PDF page 4 | lines 28-35

• Accountability
o Ensuring an entity’s action is traceable uniquely to that entity.
o Ascertains the responsibility of an entity (like a person) for its actions and decisions.
o All relevant activities events and operations on a system, e.g., failed and successful authentication attempts, are recorded in a log.

Question sources

Computer Science MCQ practice collection · Question 539

Question 83 | AM0540

What is a race condition vulnerability?

A. A vulnerability that allows unauthorized access to a system's memory

B. A vulnerability that allows attackers to intercept network traffic

C. A vulnerability that allows attackers to execute arbitrary code

D. A vulnerability that occurs when the timing of program execution leads to unexpected behavior

Hint

Trace timing over shared state or distinguish finding a weakness from demonstrating consequences.

Answer D: Timing/interleaving of shared state causes incorrect behavior

SEC27 defines a race through timing-dependent correctness over shared state, such as check-then-use or lost updates. D states that mechanism. Unauthorized memory access, interception and arbitrary code execution may be possible harms from many vulnerabilities, but they do not define the interleaving dependency of a race condition.

Why each choice is right or wrong

A - Incorrect: Memory-access harm alone does not establish a shared-state timing or interleaving dependency.

B - Incorrect: Traffic interception is a different mechanism and does not define a race.

C - Incorrect: Code execution is a possible consequence of various flaws, not the timing-based race definition.

D - Correct: Timing-dependent unexpected behavior over shared state is the defining race-condition mechanism.

Primary note proof

Study-note section SEC27

> A race occurs when correctness depends on timing/interleaving over shared state. In a check-then-open file operation, another actor may replace the file between the check and use. In a balance update, two workers can both read100 and each write90, losing one debit. Protect the whole invariant with suitable atomic operations, locks or database transactions; locking an unrelated variable does not help.

Study-note section SEC27

> Host/network/database scanners see different exposures and have false results.

Study-note section SEC27

> Static analysis inspects code, dynamic testing executes it, and fuzzing varies inputs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Computer Science MCQ practice collection · Question 540

Question 84 | AM0541

What is the principle of least privilege in program security?

A. The principle of giving users the minimum necessary permissions to perform their tasks

B. The principle of encrypting all data stored in a system

C. The principle of regularly updating software to patch vulnerabilities

D. The principle of blocking all incoming network connections by default

Hint

Identify subject, object and permitted operation before comparing policy models.

Answer A: The principle of giving users the minimum necessary permissions to perform their tasks

PDF p.5 says not to give more privileges than absolutely necessary for the required job. This directly selects the minimum-permissions option.

Why each choice is right or wrong

A - Correct: Minimum required permissions exactly matches least privilege. See PDF p.5.

B - Incorrect: Encrypting all data is a separate data-protection measure. See PDF p.5.

C - Incorrect: Patching repairs vulnerabilities rather than minimizing permissions. See PDF p.5.

D - Incorrect: Default blocking is not the definition of task-based minimum privilege. See PDF p.5.

Primary note proof

Study-note section SEC09

Least privilege gives only needed rights, for needed time. Separation of duties splits sensitive steps across parties. Deny by default, mediate every access and revoke stale permissions.

Allowing SELECT but denying UPDATE is an access-control decision that helps integrity; it does not by itself enforce all valid business-state transitions.

Study-note section SEC23

Stateless packet filtering matches headers such as source/destination, protocol and ports.

Study-note section SEC23

Rules should be explicit, ordered as implemented, least-privilege and logged appropriately.

Related reading (context, not primary note proof)

Study-note section SEC25

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 2-10

• Principles of Least Privilege: —Do not give any more privileges than absolutely necessary to do/perform the required job‖.
• Defense in Depth: having more than one layer or type of defense
• Minimization: —do not run any software, or applications that are not strictly required to do the entrusted job.‖
• Compartmentalization: limits the damage in other compartments when one compartment is compromised.

Question sources

Computer Science MCQ practice collection · Question 541

Question 85 | AM0542

What is the purpose of code obfuscation in program security?

A. To enhance program performance and execution speed

B. To make the code more difficult to understand and analyze

C. To encrypt sensitive data stored in the program

D. To detect and block malicious code during program execution

Hint

Separate observation from blocking and sensor placement from detection method.

Answer B: Make code harder to understand and analyze

SEC24 describes obfuscation as complicating understanding and analysis. It does not reliably hide embedded secrets or remove defects, so it is not equivalent to encryption or malware prevention. Nor does it imply better execution performance. B states the intended analysis-resistance purpose while preserving the note's limits.

Why each choice is right or wrong

A - Incorrect: Obfuscation is not defined as a performance optimization and need not improve speed.

B - Correct: Making code more difficult to understand/analyze is the explicit obfuscation purpose.

C - Incorrect: Obfuscation is not encryption and does not reliably protect secrets embedded in a program.

D - Incorrect: Runtime malicious-code detection/blocking is a separate defensive function rather than obfuscation.

Primary note proof

Study-note section SEC24

> A true positive is a real attack detected, false negative is a missed attack, false positive is innocent activity flagged, and true negative is innocent activity accepted. Recall=TP/(TP+FN); precision=TP/(TP+FP). Detecting 90 of 100 spam messages with five innocent messages flagged gives recall90%, precision90/95≈94.7%. Obfuscation makes code harder to understand and can hamper analysis, but does not reliably protect embedded secrets or remove vulnerabilities.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 6 | lines 2-12

> o Encipher (encrypt) - the process of converting
> plaintext to ciphertext
> o Decipher (decrypt) - recovering plaintext from
> ciphertext
> o Cryptography - the study of encryption
> principles/methods
> o Cryptanalysis (codebreaking) - the study of
> principles/ methods of breaking ciphertext without
> knowing key
> o Cryptology - the field of study which deals with both
> cryptography and cryptanalysis

Computer Security module | PDF page 10 | lines 56-62

> ✓ Vulnerability Assessment: the process of scanning
> (locating & reporting) vulnerabilities to provide a
> way to detect and resolve security problems by
> ranking the vulnerabilities.

✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Computer Science MCQ practice collection · Question 542

Question 86 | AM0543

What is a security patch in program security?

A. A technique used to prevent buffer overflow vulnerabilities

B. A software update released to fix known security vulnerabilities

C. A hardware device used to monitor network traffic for malicious activities

D. A set of security policies implemented by an organization

Hint

Identify the unsafe operation or input boundary rather than only a possible consequence.

Answer B: A software update released to fix known security vulnerabilities

The module warns that failing to update operating systems/applications leaves software vulnerable to bugs already identified and patched (PDF p.3). A security patch is the corrective software update described by B.

Why each choice is right or wrong

A - Incorrect: Buffer overflow is only one possible weakness; this is not the general update definition. See PDF p.3.

B - Correct: A corrective software update for known weaknesses follows the module's patch/update account. See PDF p.3.

C - Incorrect: A patch is a software correction, not network-monitoring hardware. See PDF p.3.

D - Incorrect: Organizational policy is not the software bug correction discussed. See PDF p.3.

Primary note proof

Study-note section SEC25

Validate expected type, length, range and structure at trust boundaries. Server validation is essential because clients are controllable. Encode output for its destination; validation is not a universal substitute for query parameterization or HTML-context encoding. Security patches repair known weaknesses; a zero-day may lack a vendor fix, but segmentation and other mitigations can still help.

Study-note section SEC33

A security policy states responsibilities and rules; a procedure supplies the steps.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 53-59

> • Example: If a company has antivirus software but does not use an anti-virus and keep the virus signatures up-to-date, this is vulnerability. The company is vulnerable to virus attacks. Similarly, if you fail to routinely update your operating systems or application software, these will remain vulnerable to software problems ("bugs") that have been identified and patched

Question sources

Computer Science MCQ practice collection · Question 543

Question 87 | AM0544

What are the basic requirements for a trusted operating system?

A. Authentication, encryption, and firewall protection

B. User-friendly interface, multitasking, and device driver support

C. Confidentiality, integrity, availability, and accountability

D. User access control, software updates, and system backup

Hint

Separate security-policy enforcement and assurance from ordinary OS features.

Answer C: Confidentiality, integrity, availability and accountability

SEC28 connects trust to enforcement of a stated policy through identity, isolation, access enforcement, protected auditing and safe recovery. Those mechanisms support the broad confidentiality, integrity, availability and accountability requirements in C. The other options list selected mechanisms or ordinary usability features; they do not state the overall security requirements as completely.

Why each choice is right or wrong

A - Incorrect: Authentication/encryption/firewalls are selected controls and do not cover the overall trust requirements.

B - Incorrect: A GUI and multitasking do not establish security trust, as SEC28 explicitly warns.

C - Correct: These broad protection/accountability goals are supported by the trusted OS enforcement and auditing mechanisms.

D - Incorrect: Access controls, updates and backups contribute mechanisms but omit the complete high-level requirement set.

Primary note proof

Study-note section SEC28

A trusted OS is relied on to enforce a stated security policy. Relevant mechanisms include identification/authentication, process and memory isolation, controlled privileges, object access enforcement, protected auditing, safe resource reuse, trusted boot/update paths and recovery behavior. The trusted computing base comprises components whose failure can violate policy; smaller, analyzable boundaries simplify assurance.

Study-note section SEC28

A GUI or multitasking support alone does not establish trust.

Study-note section SEC01

Accountability links actions to responsible entities through reliable identity and protected logs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 2 | lines 84-90

• Computer Security: (W. Stallings)
o Protection to an automated information system in order to attain the applicable objectives of preserving the confidentiality, integrity, and availability (CIA) of information system resources (includes hardware, software, firmware, information/data, and telecommunications).

Computer Security module | PDF page 4 | lines 28-35

• Accountability
o Ensuring an entity’s action is traceable uniquely to that entity.
o Ascertains the responsibility of an entity (like a person) for its actions and decisions.
o All relevant activities events and operations on a system, e.g., failed and successful authentication attempts, are recorded in a log.

Computer Security module | PDF page 11 | lines 74-79

• Most OS authentication is handled through user names and passwords. Biometric (e.g. voice, face, retina, iris, fingerprint) and physical token-based (swipe cards, pin-generating cards) authentication are sometimes used to augment simple passwords, but the costs and accuracy of the technology limit their adoption.

Question sources

Computer Science MCQ practice collection · Question 544

Question 88 | AM0545

What is the purpose of independent evaluation in computer security?

A. To certify that a system is immune to all security threats

B. To assess the effectiveness of security controls and mechanisms

C. To perform regular system backups and data recovery tests

D. To ensure compliance with industry standards and regulations

Hint

Distinguish requirements, technical evaluation, evidence and certification decisions.

Answer B: Assess security controls against stated claims

Independent evaluation examines evidence and tests controls against the product's specified functional and assurance claims. B captures that assessment purpose. SEC29 explicitly rejects immunity to all attacks and universal regulatory conclusions. Backups are an operational recovery practice rather than the evaluation's defining task.

Why each choice is right or wrong

A - Incorrect: A scoped evaluation cannot certify immunity to every present and future security threat.

B - Correct: Examining evidence and testing controls assesses whether their stated security claims are met.

C - Incorrect: Routine backups and restore tests are operational recovery tasks, not the defining independent-evaluation purpose.

D - Incorrect: Evaluation assesses specified security claims; it does not automatically establish every industry/regulatory obligation.

Primary note proof

Study-note section SEC29

> Under Common Criteria, define the Target of Evaluation (TOE), its configuration and environment. A Security Target (ST) states threats, assumptions, objectives, functional requirements and assurance requirements; a Protection Profile describes reusable requirements for a product category. Evaluators examine developer evidence, test and analyze vulnerabilities against those claims. A certification body oversees/reviews the evaluation and issues the certification decision; sponsors commission work and developers supply product/evidence.

Study-note section SEC29

> A certificate applies to a specified product/configuration and claims, not immunity to all future attacks.

Study-note section SEC29

An Evaluation Assurance Level packages assurance requirements; EAL1-7 express increasing evaluation rigor, not a universal ranking of security functions.

Related reading (context, not primary note proof)

Study-note section SEC33

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Computer Security module | PDF page 10 | lines 85-90

o Penetration Test (pen test):
✓ To identify both weaknesses (vulnerabilities) & strengths, to enable a full risk assessment.
✓ Determine whether a system is vulnerable, defenses were sufficient and which defenses (if any) the test defeated.

Question sources

Computer Science MCQ practice collection · Question 545

Question 89 | AM0546

What are the evaluation criteria used in the independent evaluation of operating systems?

A. User interface design and performance metrics

B. Compliance with international security standards

C. Vulnerability scanning and penetration testing results

D. Security functionality, assurance, and vulnerabilities

Hint

Distinguish requirements, technical evaluation, evidence and certification decisions.

Answer D: Security functionality, assurance and vulnerability analysis

SEC29 gives functional and assurance requirements in the Security Target, along with evaluator vulnerability analysis. D captures those evaluation dimensions. User-interface/performance measures are not the security criteria at issue; standards compliance is too broad, and scan/penetration results are evidence rather than the complete requirements framework.

Why each choice is right or wrong

A - Incorrect: Interface design and generic performance metrics do not state the security functional/assurance criteria.

B - Incorrect: Generic standards compliance omits the scoped Security Target's concrete functionality and assurance requirements.

C - Incorrect: Testing results can be evidence but do not replace the requirements and assurance criteria themselves.

D - Correct: Functionality, assurance and vulnerability analysis match the evaluation dimensions described in SEC29.

Primary note proof

Study-note section SEC29

> Under Common Criteria, define the Target of Evaluation (TOE), its configuration and environment. A Security Target (ST) states threats, assumptions, objectives, functional requirements and assurance requirements; a Protection Profile describes reusable requirements for a product category. Evaluators examine developer evidence, test and analyze vulnerabilities against those claims. A certification body oversees/reviews the evaluation and issues the certification decision; sponsors commission work and developers supply product/evidence.

Study-note section SEC27

> Vulnerability assessment finds and prioritizes weaknesses; authorized penetration testing attempts to demonstrate consequences.

Study-note section SEC27

> Static analysis inspects code, dynamic testing executes it, and fuzzing varies inputs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

> ✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
> ✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Computer Security module | PDF page 10 | lines 85-90

o Penetration Test (pen test):
✓ To identify both weaknesses (vulnerabilities) & strengths, to enable a full risk assessment.
✓ Determine whether a system is vulnerable, defenses were sufficient and which defenses (if any) the test defeated.

Question sources

Computer Science MCQ practice collection · Question 546

Question 90 | AM0547

What is the evaluation process for trusted operating systems?

A. Development, testing, and deployment

B. Risk assessment, vulnerability scanning, and patching

C. Security requirements analysis, design, and implementation

D. Independent evaluation, testing, and certification

Hint

Distinguish requirements, technical evaluation, evidence and certification decisions.

Answer D: Independent evaluation, testing and certification

SEC29 separates developer product/evidence work, evaluator examination/testing and the certification body's review/decision. D names that evaluation-and-certification process. Development/deployment, patching and security design are related lifecycle activities but do not themselves constitute independent examination and the subsequent scoped certification decision.

Why each choice is right or wrong

A - Incorrect: Development/testing/deployment is a general software lifecycle and omits independent certification review.

B - Incorrect: Risk assessment and patching are administration/security operations rather than the full independent evaluation process.

C - Incorrect: Requirements/design/implementation are development activities that supply evidence, not the independent evaluation decision.

D - Correct: Independent examination/testing followed by certification review matches the process and roles in SEC29.

Primary note proof

Study-note section SEC29

Under Common Criteria, define the Target of Evaluation (TOE), its configuration and environment. A Security Target (ST) states threats, assumptions, objectives, functional

requirements and assurance requirements; a Protection Profile describes reusable requirements for a product category. Evaluators examine developer evidence, test and analyze vulnerabilities against those claims. A certification body oversees/reviews the evaluation and issues the certification decision; sponsors commission work and developers supply product/evidence.

Study-note section SEC29

An Evaluation Assurance Level packages assurance requirements; EAL1-7 express increasing evaluation rigor, not a universal ranking of security functions.

Study-note section SEC29

CCRA is a mutual-recognition arrangement, not a single worldwide testing laboratory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Computer Science MCQ practice collection · Question 547

Question 91 | AM0548

Under Common Criteria, which role oversees and reviews an independent product evaluation and issues the certification decision?

A. International Organization for Standardization (ISO)

B. National Security Agency (NSA)

C. National Institute of Standards and Technology (NIST)

D. The certification body of the applicable evaluation scheme

Hint

Distinguish requirements, technical evaluation, evidence and certification decisions.

Answer D: The certification body of the applicable evaluation scheme

Evaluators examine evidence, test the target and analyze vulnerabilities against its stated claims. The applicable scheme's certification body oversees or reviews that evaluation and issues the certification decision. CCRA concerns mutual recognition rather than acting as a worldwide testing laboratory. The answer names the responsible scheme role instead of assigning a universal responsibility to one national organization without a jurisdiction or scheme being specified.

Why each choice is right or wrong

A - Incorrect: ISO's name alone does not identify the applicable scheme certification body.

B - Incorrect: NSA's name alone does not identify the applicable scheme certification body.

C - Incorrect: NIST's name alone does not identify the applicable scheme certification body.

D - Correct: This is the oversight and certification-decision role specified here.

Primary note proof

Study-note section SEC29

> Under Common Criteria, define the Target of Evaluation (TOE), its configuration and environment. A Security Target (ST) states threats, assumptions, objectives, functional requirements and assurance requirements; a Protection Profile describes reusable requirements for a product category. Evaluators examine developer evidence, test and analyze vulnerabilities against those claims. A certification body oversees/reviews the evaluation and issues the certification decision; sponsors commission work and developers supply product/evidence.

Study-note section SEC29

> CCRA is a mutual-recognition arrangement, not a single worldwide testing laboratory.

Study-note section SEC29

> A certificate applies to a specified product/configuration and claims, not immunity to all future attacks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

> ✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
> ✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Computer Science MCQ practice collection · Question 548

Question 92 | AM0549

What is the goal of security functionality evaluation in trusted operating systems?

A. To assess the system's resistance to external attacks

B. To evaluate the system's ability to protect sensitive information

C. To analyze the system's performance and resource utilization

D. To identify potential vulnerabilities and weaknesses in the system

Hint

Separate security-policy enforcement and assurance from ordinary OS features.

Answer B: Evaluate claimed protection of sensitive information

Functionality evaluation addresses whether claimed security behavior performs its intended protection, while assurance concerns confidence in that implementation. B most directly states that protection capability. External-attack resistance is narrower, performance is not security functionality, and weakness discovery is the vulnerability-analysis aspect rather than the complete functional goal.

Why each choice is right or wrong

A - Incorrect: Resistance to external attacks is only a subset and omits internal threats and broader claimed security behavior.

B - Correct: Assessing the ability to protect sensitive information matches evaluation of claimed security functionality.

C - Incorrect: Generic performance/resource use is not the security-functionality goal described.

D - Incorrect: Weakness identification is vulnerability analysis, a related evaluation activity rather than the overall functionality goal.

Primary note proof

Study-note section SEC28

> A trusted OS is relied on to enforce a stated security policy. Relevant mechanisms include identification/authentication, process and memory isolation, controlled privileges, object access enforcement, protected auditing, safe resource reuse, trusted boot/update paths and recovery behavior. The trusted computing base comprises components whose failure can violate policy; smaller, analyzable boundaries simplify assurance.

Study-note section SEC29

> A Security Target (ST) states threats, assumptions, objectives, functional requirements and assurance requirements; a Protection Profile describes reusable requirements for a product category.

Study-note section SEC28

Assurance is justified confidence that controls implement the claims; functionality is what controls claim to do.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 2 | lines 84-90

• Computer Security: (W. Stallings)
o Protection to an automated information system in order to attain the applicable objectives of preserving the confidentiality, integrity, and availability (CIA) of information system resources (includes hardware, software, firmware, information/data, and telecommunications).

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Computer Science MCQ practice collection · Question 549

Question 93 | AM0550

What is the assurance level in the evaluation of trusted operating systems?

A. The level of protection provided by the system's security controls

B. The level of confidence in the system's security claims and implementation

C. The level of user satisfaction with the system's performance

D. The level of compliance with industry standards and regulations

Hint

Separate security-policy enforcement and assurance from ordinary OS features.

Answer B: Confidence in security claims and implementation

Assurance describes justified confidence that controls implement their claims; SEC28 states that distinction and SEC29 describes increasing evaluation rigor. B matches confidence in implementation. It must not be confused with the number or strength of functions, user satisfaction or generic compliance. Products with different scopes cannot be ranked automatically by assurance number alone.

Why each choice is right or wrong

A - Incorrect: Protection functionality is what controls do; assurance is confidence in the evidence and implementation.

B - Correct: Assurance is justified confidence in security claims and their implementation.

C - Incorrect: User satisfaction with performance is not security assurance.

D - Incorrect: Generic compliance is not the defined confidence measure or evaluation rigor represented by assurance.

Primary note proof

Study-note section SEC28

> A trusted OS is relied on to enforce a stated security policy. Relevant mechanisms include identification/authentication, process and memory isolation, controlled privileges, object access enforcement, protected auditing, safe resource reuse, trusted boot/update paths and recovery behavior. The trusted computing base comprises components whose failure can violate policy; smaller, analyzable boundaries simplify assurance.

Study-note section SEC28

> Assurance is justified confidence that controls implement the claims; functionality is what controls claim to do.

Study-note section SEC29

> An Evaluation Assurance Level packages assurance requirements; EAL1-7 express increasing evaluation rigor, not a universal ranking of security functions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 2 | lines 77-83

> • IA: measures that protect and defend information and ISs by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation.
> • Information protection or InfoSec: protection of information and ISs from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction.

Computer Security module | PDF page 10 | lines 56-62

> ✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
> ✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Question sources

Computer Science MCQ practice collection · Question 550

Question 94 | AM0551

What is the role of vulnerability analysis in the independent evaluation of operating systems?

A. To identify potential weaknesses and vulnerabilities in the system

B. To analyze the system's performance and resource utilization

C. To assess the system's resistance to external attacks

D. To evaluate the system's ability to protect sensitive information

Hint

Trace timing over shared state or distinguish finding a weakness from demonstrating consequences.

Answer A: To identify potential weaknesses and vulnerabilities in the system

Vulnerability assessment explicitly locates and reports vulnerabilities and pen testing identifies weaknesses/strengths (PDF p.10). Thus identifying weaknesses is the directly supported purpose of vulnerability analysis, including when used within an evaluation.

Why each choice is right or wrong

A - Correct: Locating weaknesses directly matches the stated assessment task. See PDF p.10.

B - Incorrect: Performance/resource utilization is not the stated vulnerability-finding task. See PDF p.10.

C - Incorrect: Attack resistance is a broader test concern; locating weaknesses is the direct analysis function asked. See PDF p.10.

D - Incorrect: Protecting sensitive information is the broader security goal, not the immediate vulnerability-analysis output. See PDF p.10.

Primary note proof

Study-note section SEC27

> Vulnerability assessment finds and prioritizes weaknesses; authorized penetration testing attempts to demonstrate consequences. Static analysis inspects code, dynamic testing executes it, and fuzzing varies inputs. Host/network/database scanners see different exposures and have false results. Correct findings, patch, test and recheck; testing is not proof of absence. Build threat modeling, review, least privilege, dependency management and safe error handling into development.

Study-note section SEC29

> An Evaluation Assurance Level packages assurance requirements; EAL1-7 express increasing evaluation rigor, not a universal ranking of security functions.

Study-note section SEC29

> Evaluators examine developer evidence, test and analyze vulnerabilities against those claims.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

> ✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
> ✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Computer Security module | PDF page 10 | lines 85-90

> o Penetration Test (pen test):
> ✓ To identify both weaknesses (vulnerabilities) & strengths, to enable a full risk assessment.
> ✓ Determine whether a system is vulnerable, defenses were sufficient and which defenses (if any) the test defeated.

Question sources

Computer Science MCQ practice collection · Question 551

Question 95 | AM0552

What is the purpose of certification in the evaluation of trusted operating systems?

A. To ensure compliance with industry standards and regulations

B. To guarantee absolute security and protection against all threats

C. To provide a warranty for the system's performance and reliability

D. To demonstrate that the system meets specific security requirements

Hint

Distinguish requirements, technical evaluation, evidence and certification decisions.

Answer D: Demonstrate conformance to specified security requirements

Certification records a reviewed decision about a scoped product/configuration meeting specified claims. D fits that purpose. The decision is not an unlimited guarantee against attacks, a performance warranty or proof of every industry/regulatory requirement. SEC29's scope qualification is essential to interpreting what the certificate actually establishes.

Why each choice is right or wrong

A - Incorrect: Generic industry/regulatory compliance exceeds the specific scoped security claims established by certification.

B - Incorrect: Certification does not guarantee absolute security or protection against all threats.

C - Incorrect: A security certificate is not a warranty of general performance and reliability.

D - Correct: Certification demonstrates satisfaction of specified security requirements within the evaluated scope/configuration.

Primary note proof

Study-note section SEC29

> Under Common Criteria, define the Target of Evaluation (TOE), its configuration and environment. A Security Target (ST) states threats, assumptions, objectives, functional requirements and assurance requirements; a Protection Profile describes reusable requirements for a product category. Evaluators examine developer evidence, test and analyze vulnerabilities against those claims. A certification body oversees/reviews the evaluation and issues the certification decision; sponsors commission work and developers supply product/evidence.

Study-note section SEC29

> A certificate applies to a specified product/configuration and claims, not immunity to all future attacks.

Study-note section SEC29

> A higher number does not make a product with different scope automatically safer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

> ✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
> ✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Computer Security module | PDF page 10 | lines 85-90

> o Penetration Test (pen test):
> ✓ To identify both weaknesses (vulnerabilities) & strengths, to enable a full risk assessment.
> ✓ Determine whether a system is vulnerable, defenses were sufficient and which defenses (if any) the test defeated.

Question sources

Computer Science MCQ practice collection · Question 552

Question 96 | AM0553

What is the significance of independent evaluation in trusted operating systems?

A. It guarantees absolute security and protection against all threats.

B. It ensures compliance with industry standards and regulations.

C. It provides assurance that the system is immune to all vulnerabilities.

D. It increases confidence in the system's security and reliability.

Hint

Distinguish requirements, technical evaluation, evidence and certification decisions.

Answer D: Increase justified confidence in security and reliability

Independent examination and testing add evidence supporting confidence in stated security behavior and reliability within scope. D gives that qualified benefit. SEC29 rejects the inference that certification makes a product immune to future attacks; it also does not establish every unrelated regulatory obligation. Confidence increases through evidence, rather than becoming an absolute guarantee.

Why each choice is right or wrong

A - Incorrect: Independent evaluation cannot guarantee absolute security against every threat.

B - Incorrect: It does not automatically establish compliance with all industry standards and regulations.

C - Incorrect: Tests and evidence cannot prove immunity to every possible vulnerability.

D - Correct: Independent scrutiny increases justified confidence in the product’s scoped security and dependable operation.

Primary note proof

Study-note section SEC29

> Under Common Criteria, define the Target of Evaluation (TOE), its configuration and environment. A Security Target (ST) states threats, assumptions, objectives, functional requirements and assurance requirements; a Protection Profile describes reusable requirements for a product category. Evaluators examine developer evidence, test and analyze vulnerabilities against those claims. A certification body oversees/reviews the evaluation and issues the certification decision; sponsors commission work and developers supply product/evidence.

Study-note section SEC29

> A certificate applies to a specified product/configuration and claims, not immunity to all future attacks.

Study-note section SEC29

A higher number does not make a product with different scope automatically safer.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 10 | lines 56-62

✓ Vulnerability Assessment: the process of scanning (locating & reporting) vulnerabilities to provide a way to detect and resolve security problems by ranking the vulnerabilities.
✓ Penetration Testing: an authorized simulated attack on a computer system, performed to evaluate the security of the system.

Computer Security module | PDF page 10 | lines 85-90

o Penetration Test (pen test):
✓ To identify both weaknesses (vulnerabilities) & strengths, to enable a full risk assessment.
✓ Determine whether a system is vulnerable, defenses were sufficient and which defenses (if any) the test defeated.

Question sources

Computer Science MCQ practice collection · Question 553

Question 97 | AM0554

Which listed properties are fundamental database-security goals, rather than operational growth or transaction-execution capabilities? Select all that apply.

A. Availability

B. Concurrency

C. Scalability

D. Confidentiality

Hint

Match the control's scope to the disclosure or access path.

Answer A, D: Availability; Confidentiality

Database security includes confidentiality, integrity, availability and accountability together. Availability and confidentiality are both present in the choices and must be selected. Concurrency concerns overlapping transactions and can require integrity-preserving control, while scalability concerns capacity growth. These capabilities matter to a database system but do not replace the named security goals. The revised selection wording preserves both valid goals instead of choosing confidentiality arbitrarily.

Why each choice is right or wrong

A - Correct: Availability is explicitly part of database security.

B - Incorrect: Concurrency is an execution capability, not a named security goal in this taxonomy.

C - Incorrect: Scalability describes growth capability rather than a named security goal.

D - Correct: Confidentiality is explicitly part of database security.

Primary note proof

Study-note section SEC30

> Database security needs confidentiality, integrity, availability and accountability together. Authenticate connections, give roles only necessary table/column/row privileges, restrict views and audit sensitive changes. A user allowed to read a report need not read the underlying full table. Check application-level authorization as well as database rights.

Study-note section SEC01

> One event can harm several goals.

Study-note section SEC01

> Information assurance includes confidence, risk management, protection, detection, response and restoration across the information lifecycle.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 2 | lines 84-90

> • Computer Security: (W. Stallings)
> o Protection to an automated information system in order to attain the applicable objectives of preserving the confidentiality, integrity, and availability (CIA) of information system resources (includes hardware, software, firmware, information/data, and telecommunications).

Computer Security module | PDF page 3 | lines 71-78

> • Confidentiality: Data confidentiality and Privacy
> o is the prevention of unauthorized disclosure of information.

o In other words, confidentiality means keeping information private or safe.
o it may be important for military, business or personal reasons.
o it may also be known as privacy or secrecy.

Computer Security module | PDF page 3 | lines 87-91

• Availability:
o Assures that systems work promptly and service is not denied to authorized users.
o Information should be accessible and usable upon appropriate demand by an authorized user.

Question sources

Computer Science MCQ practice collection · Question 554

Question 98 | AM0555

Which technique ensures the reliability of a database system?

A. Data encryption

B. Data replication

C. Role-based access control

D. Intrusion detection system

Hint

Separate valid data states, permission to update and recovery.

Answer B: Data replication can improve service availability

Among these options replication supplies alternate service capacity, supporting database reliability/availability when failover and failure domains are properly handled. It does not guarantee correctness or replace backups; malicious changes can replicate too. Encryption, RBAC and IDS serve confidentiality, access restriction and detection rather than redundant database service capacity.

Why each choice is right or wrong

A - Incorrect: Encryption primarily protects confidentiality rather than providing alternate database service capacity.

B - Correct: Replication can supply alternate service capacity, with reliability dependent on failover and independent failure handling.

C - Incorrect: RBAC controls permitted operations; it does not itself provide redundant database capacity or crash recovery.

D - Incorrect: An IDS detects suspicious activity and does not itself maintain database availability after component failure.

Primary note proof

Study-note section SEC31

> Write-ahead logging persists recovery records before dependent data-file changes, supporting recovery after a crash. Backups, restore tests and separately protected logs recover from deletion/corruption; replication provides alternate service capacity but does not replace backups. Redundancy improves availability only with appropriate failover and independent failure handling. Distinguish confidentiality controls from reliability controls and from integrity constraints.

Study-note section SEC30

> Strong access controls, encryption and protected transmission can all contribute to secrecy; compare their roles and assumptions.

Study-note section SEC30

> Replication is useful for availability but also copies malicious updates unless separate recovery controls exist.

Related reading (context, not primary note proof)

Study-note section SEC09

Study-note section SEC24

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 87-91

> • Availability:
> o Assures that systems work promptly and service is not
> denied to authorized users.
> o Information should be accessible and usable upon
> appropriate demand by an authorized user.

Computer Security module | PDF page 7 | lines 48-54

> Symmetric key cryptography
> • Uses conventional / secret-key / single-key.
> • The secret key shared by both sender and receiver.
> • All classical encryption algorithms are private-key.
> o Use the same key (shared key) for encryption and
> decryption process.
> • If this key is disclosed communications are compromised.

Computer Security module | PDF page 11 | lines 80-87

> 6. Access control
> • Since many systems (such as router access control lists)

define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Computer Security module | PDF page 4 | lines 89-99

o Encryption: primary control that protects CIA, by which Cleartext scambled into ciphertext.
o Software controls: password checker, virus scanner, IDS (intrusion detection system)
o Hardware controls: provide higher degree of security; Locks, Smart cards, dongles, hadware keys...
o Policies and procedures: Alignment with users' legal and ethical standards; eg. password policy
o Physical controls: natural and man-made disaster protection; walls, locks, guards, security, cameras, backup copies and archives

Question sources

Computer Science MCQ practice collection · Question 555

Question 99 | AM0556

Which option gives the most direct definition of database integrity as a property of stored data, rather than a broader recovery activity, confidentiality control or monitoring activity?

A. Ensuring data consistency and accuracy

B. Protecting data from unauthorized access

C. Preventing data loss or corruption

D. Monitoring and detecting security breaches

Hint

Separate valid data states, permission to update and recovery.

Answer A: Ensuring data consistency and accuracy

Integrity concerns valid, accurate and consistent data and valid state transitions. A states that property directly. Preventing corruption can contribute to integrity, but the combined loss-or-corruption wording also includes availability and recovery concerns, so it is a broader activity rather than the most direct definition requested. Access protection can serve confidentiality, and monitoring detects events; neither describes the data-validity property as precisely as A.

Why each choice is right or wrong

A - Correct: Accuracy and consistency directly characterize the requested integrity property.

B - Incorrect: Generic access protection is broader and does not directly define data validity.

C - Incorrect: Preventing loss or corruption is a broader protection/recovery activity, though preventing corruption contributes to integrity.

D - Incorrect: Monitoring is a detection activity rather than the property of valid data.

Primary note proof

Study-note section SEC31

> Integrity requires valid data and valid transitions. NOT NULL rejects missing mandatory values; UNIQUE/primary keys prevent duplicate identifiers; foreign keys preserve references; CHECK enforces declared predicates. Transactions group a transfer's debit and credit atomically; concurrency control prevents unsafe interleavings. Roles/ACLs limit who may update, but constraints and transaction logic determine whether an authorized update is valid.

Study-note section SEC01

> Information assurance includes confidence, risk management, protection, detection, response and restoration across the information lifecycle.

Study-note section SEC01

> Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 79-86

> • Integrity: covers Data integrity and System integrity
> o the prevention of unauthorized writing or modification of information.
> o Integrity in a computer system means that there is an external consistency in the system - everything is as it is expected to be.
> o Data integrity: the data stored on the computer is the same as what is intended.

Computer Security module | PDF page 3 | lines 87-91

> • Availability:
> o Assures that systems work promptly and service is not denied to authorized users.
> o Information should be accessible and usable upon appropriate demand by an authorized user.

Question sources

Computer Science MCQ practice collection · Question 556

Question 100 | AM0557

Which controls directly restrict disclosure of database contents in their applicable scopes, rather than primarily reviewing evidence after activity? Select all that apply.

A. Using strong access controls

B. Implementing encryption techniques

C. Enforcing secure data transmission protocols

D. Conducting regular security audits

Hint

Match the control's scope to the disclosure or access path.

Answer A, B, C: Using strong access controls; Implementing encryption techniques; Enforcing secure data transmission protocols

Access controls restrict which subjects can read data. Encryption at rest protects stolen storage when keys remain separate, and protected transmission secures data on the connection. All three can directly contribute to secrecy in their respective scopes, though none prevents every disclosure. Audits principally inspect evidence and reveal problems rather than themselves concealing contents or enforcing every access. The stated direct-control distinction makes A, B and C the valid selection.

Why each choice is right or wrong

A - Correct: Access rules can directly restrict unauthorized readers.

B - Correct: Encryption protects content in its applicable scope when keys are properly separated.

C - Correct: Protected transport restricts disclosure on the communication channel.

D - Incorrect: Audits primarily review evidence rather than directly concealing content or enforcing each read.

Primary note proof

Study-note section SEC30

> Database security needs confidentiality, integrity, availability and accountability together. Authenticate connections, give roles only necessary table/column/row privileges, restrict views and audit sensitive changes. A user allowed to read a report need not read the underlying full table. Check application-level authorization as well as database rights.

Study-note section SEC30

> Encryption at rest protects stolen storage/backups when keys remain separate; transport encryption protects the connection.

Study-note section SEC30

Strong access controls, encryption and protected transmission can all contribute to secrecy; compare their roles and assumptions.

Related reading (context, not primary note proof)

Study-note section SEC02

Study-note section SEC09

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 71-78

• Confidentiality: Data confidentiality and Privacy
o is the prevention of unauthorized disclosure of information.
o In other words, confidentiality means keeping information private or safe.
o it may be important for military, business or personal reasons.
o it may also be known as privacy or secrecy.

Computer Security module | PDF page 4 | lines 15-27

• Authentication
o is proving a claim - usually that you are who you say you are.
o for example it may be obtained by the provision of a password or by a scan of your retina.
• Access controls
o provide the limitation and control of access to authorized users through identification and authentication.
o A system needs to be able to identify and authenticate users for access to data, applications and hardware.
o In a large system there may be a complex structure determining which users and applications have access to which objects.

Computer Security module | PDF page 7 | lines 48-54

Symmetric key cryptography
• Uses conventional / secret-key / single-key.
• The secret key shared by both sender and receiver.
• All classical encryption algorithms are private-key.
o Use the same key (shared key) for encryption and decryption process.
• If this key is disclosed communications are compromised.

Computer Security module | PDF page 11 | lines 2-7

o A cryptographic protocol for operating NW services securely over an unsecured NW.
o Best-known application: for remote login to computer systems by users.
o Common applications: remote command-line login and execution.

Question sources

Computer Science MCQ practice collection · Question 557

Question 101 | AM0558

What is inference control in database security?

A. Controlling the flow of information within a database

B. Enforcing access controls based on user roles

C. Preventing unauthorized data leakage through inference

D. Detecting and mitigating security breaches in real-time

Hint

Consider what can be inferred from allowed outputs as well as direct access.

Answer C: Prevent protected facts being inferred from allowed outputs

Inference control addresses indirect disclosure, such as subtracting a known salary from a two-person total to learn the protected value. SEC32 describes restricting/coarsening outputs and tracking queries to reduce that leakage. C names the specific purpose, while generic information flow, roles and real-time detection do not identify inference from combinations of permitted results.

Why each choice is right or wrong

A - Incorrect: Information-flow control is broader; this option omits inference from permitted outputs and auxiliary knowledge.

B - Incorrect: Role-based access alone does not prevent inference from otherwise permitted query results.

C - Correct: Inference control limits indirect disclosure of protected facts through combinations of outputs and outside knowledge.

D - Incorrect: Real-time breach detection is a separate incident-monitoring function rather than inference control.

Primary note proof

Study-note section SEC32

Inference reveals protected facts by combining individually permitted outputs with other knowledge. If a report gives a two-person salary total and one salary is known, subtraction

reveals the other. Blocking direct SELECT of a salary column does not stop that inference. Restrict small groups and overlapping queries, track query history, coarsen/suppress results, or introduce carefully designed noise; each trades utility against leakage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 71-78

• Confidentiality: Data confidentiality and Privacy
o is the prevention of unauthorized disclosure of information.
o In other words, confidentiality means keeping information private or safe.
o it may be important for military, business or personal reasons.
o it may also be known as privacy or secrecy.

Computer Security module | PDF page 11 | lines 80-87

6. Access control
• Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Question sources

Computer Science MCQ practice collection · Question 558

Question 102 | AM0559

How can multi-level databases ensure data confidentiality?

A. By using data encryption techniques

B. By implementing mandatory access controls

C. By conducting regular security audits

D. By employing intrusion detection systems

Hint

Consider what can be inferred from allowed outputs as well as direct access.

Answer B: Mandatory label/clearance access controls

SEC32 distinguishes multilevel databases by mandatory label/clearance restrictions, including controlled read/write flows. B names that mechanism. Encryption, audits and IDS can support security, but none alone enforces the multilevel policy. Even mandatory controls need consideration of inference and covert channels rather than a blanket confidentiality guarantee.

Why each choice is right or wrong

A - Incorrect: Encryption can protect stored/transmitted content but does not itself enforce multilevel classification/clearance rules.

B - Correct: Mandatory controls compare clearances/classifications and enforce the multilevel disclosure policy.

C - Incorrect: Audits review activity and controls; they do not themselves enforce every multilevel access decision.

D - Incorrect: IDS detects suspicious activity but does not implement the database's mandatory label policy.

Primary note proof

Study-note section SEC32

> A multilevel database stores items at different labels and enforces mandatory policy using subject clearances, classifications and compartments. Bell-LaPadula-style read/write restrictions prevent direct downward confidentiality flow; trusted declassification is a separate controlled action. Views alone are not proof of multilevel isolation. Covert channels, query results and error messages may still reveal information, so evaluate both direct access and indirect inference.

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline.

Study-note section SEC30

> Strong access controls, encryption and protected transmission can all contribute to secrecy; compare their roles and assumptions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 11 | lines 80-87

> 6. Access control
> • Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
> • An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Computer Security module | PDF page 3 | lines 71-78

• Confidentiality: Data confidentiality and Privacy
o is the prevention of unauthorized disclosure of information.
o In other words, confidentiality means keeping information private or safe.
o it may be important for military, business or personal reasons.
o it may also be known as privacy or secrecy.

Question sources

Computer Science MCQ practice collection · Question 559

Question 103 | AM0560

Which security requirement is critical for protecting sensitive data in a database?

A. Availability

B. Scalability

C. Confidentiality

D. Concurrency

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer C: Confidentiality

Protecting sensitive data from disclosure requires confidentiality, which the module also calls privacy or secrecy (PDF p.3). Other system properties do not specifically express that disclosure restriction.

Why each choice is right or wrong

A - Incorrect: Availability protects timely authorized access rather than secrecy. See PDF p.3.

B - Incorrect: Scalability is not the disclosed-data protection goal. See PDF p.3.

C - Correct: Confidentiality is the stated secrecy/disclosure protection goal. See PDF p.3.

D - Incorrect: Concurrency is not the stated secrecy/disclosure protection goal. See PDF p.3.

Primary note proof

Study-note section SEC01

Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to

authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC30

Database security needs confidentiality, integrity, availability and accountability together.

Study-note section SEC01

Data security protects information at rest, in use and in transit, including accidental corruption as well as intentional attacks.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 71-78

• Confidentiality: Data confidentiality and Privacy
o is the prevention of unauthorized disclosure of information.
o In other words, confidentiality means keeping information private or safe.
o it may be important for military, business or personal reasons.
o it may also be known as privacy or secrecy.

Computer Security module | PDF page 3 | lines 87-91

• Availability:
o Assures that systems work promptly and service is not denied to authorized users.
o Information should be accessible and usable upon appropriate demand by an authorized user.

Question sources

Computer Science MCQ practice collection · Question 560

Question 104 | AM0561

What is the purpose of data encryption in database security?

A. To ensure data availability

B. To protect data integrity

C. To achieve data secrecy

D. To control data access

Hint

Match the control's scope to the disclosure or access path.

Answer C: To achieve data secrecy

The module equates confidentiality with secrecy (PDF p.3) and explains shared-key encryption so an adversary cannot understand the contents (p.7). Applying that transformation to database contents protects their secrecy.

Why each choice is right or wrong

A - Incorrect: Encryption alone does not ensure that data remains accessible on time. See PDF p.3, PDF p.7.

B - Incorrect: Plain encryption is not, by itself, the unauthorized-modification definition of integrity. See PDF p.3, PDF p.7.

C - Correct: Rendering data unintelligible without the key serves secrecy. See PDF p.3, PDF p.7.

D - Incorrect: Access-control permission decisions are a separate subject/object mechanism. See PDF p.3, PDF p.7.

Primary note proof

Study-note section SEC30

> Database security needs confidentiality, integrity, availability and accountability together. Authenticate connections, give roles only necessary table/column/row privileges, restrict views and audit sensitive changes. A user allowed to read a report need not read the underlying full table. Check application-level authorization as well as database rights.

Study-note section SEC30

> Strong access controls, encryption and protected transmission can all contribute to secrecy; compare their roles and assumptions.

Study-note section SEC30

> Encryption at rest protects stolen storage/backups when keys remain separate; transport encryption protects the connection.

Related reading (context, not primary note proof)

Study-note section SEC12

Study-note section SEC01

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 71-78

> • Confidentiality: Data confidentiality and Privacy
> o is the prevention of unauthorized disclosure of
> information.
> o In other words, confidentiality means keeping information
> private or safe.
> o it may be important for military, business or personal

reasons.
o it may also be known as privacy or secrecy.

Computer Security module | PDF page 7 | lines 48-54

Symmetric key cryptography
• Uses conventional / secret-key / single-key.
• The secret key shared by both sender and receiver.
• All classical encryption algorithms are private-key.
o Use the same key (shared key) for encryption and
decryption process.
• If this key is disclosed communications are compromised.

Question sources

Computer Science MCQ practice collection · Question 561

Question 105 | AM0562

Which permission mechanisms can prevent unauthorized database updates, while constraints and transaction logic separately enforce the validity of authorized changes? Select all that apply.

A. Access control lists

B. Data replication

C. Intrusion detection systems

D. Role-based access control

Hint

Identify subject, object and permitted operation before comparing policy models.

Answer A, D: Access control lists; Role-based access control

ACLs associate objects with allowed subjects or actions, while RBAC assigns update permissions through roles. Both can prevent updates by unauthorized subjects and therefore contribute to integrity. They do not by themselves establish that every authorized state change satisfies business rules; constraints and transaction logic address that separate requirement. Replication may copy bad updates, and intrusion detection reveals activity rather than serving as the requested permission mechanism.

Why each choice is right or wrong

A - Correct: ACLs can deny update permission to unauthorized subjects.

B - Incorrect: Replication copies state and is not an update-permission mechanism.

C - Incorrect: Intrusion detection observes or alerts rather than defining update permissions.

D - Correct: RBAC can restrict update permission through role assignments.

Primary note proof

Study-note section SEC09

Subjects request operations on objects. An ACL associates an object/resource with allowed subjects or actions; a network ACL matches traffic such as address, protocol and port. Classification and filtering are direct effects; high availability or malware detection is not guaranteed by an ACL.

Study-note section SEC31

Roles/ACLs limit who may update, but constraints and transaction logic determine whether an authorized update is valid.

Study-note section SEC09

Allowing SELECT but denying UPDATE is an access-control decision that helps integrity; it does not by itself enforce all valid business-state transitions.

Related reading (context, not primary note proof)

Study-note section SEC24

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 79-86

• Integrity: covers Data integrity and System integrity
o the prevention of unauthorized writing or modification of information.
o Integrity in a computer system means that there is an external consistency in the system - everything is as it is expected to be.
o Data integrity: the data stored on the computer is the same as what is intended.

Computer Security module | PDF page 11 | lines 80-87

6. Access control
• Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Question sources

Computer Science MCQ practice collection · Question 562

Question 106 | AM0563

What is the primary purpose of inference control in database security?

A. To prevent unauthorized data modification

B. To enforce secure data transmission

C. To detect and prevent unauthorized data leakage

D. To control data access based on user roles

Hint

Consider what can be inferred from allowed outputs as well as direct access.

Answer C: Prevent unauthorized information leakage through inference

Inference control concerns disclosure derived from combinations of permitted outputs and outside knowledge rather than direct unauthorized writes or merely role membership. SEC32's salary example shows why direct column restrictions are insufficient. C states this leakage-prevention goal; secure transmission and role-based decisions address different channels or controls.

Why each choice is right or wrong

A - Incorrect: Unauthorized modification concerns integrity, whereas inference control targets indirect disclosure.

B - Incorrect: Secure transport protects a connection but does not prevent inference from authorized query results.

C - Correct: Detecting/reducing unauthorized information leakage from combined results is the inference-control goal.

D - Incorrect: Role permissions alone do not stop users inferring protected facts from outputs they may access.

Primary note proof

Study-note section SEC32

> Inference reveals protected facts by combining individually permitted outputs with other knowledge. If a report gives a two-person salary total and one salary is known, subtraction reveals the other. Blocking direct SELECT of a salary column does not stop that inference. Restrict small groups and overlapping queries, track query history, coarsen/suppress results, or introduce carefully designed noise; each trades utility against leakage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 3 | lines 71-78

> • Confidentiality: Data confidentiality and Privacy
> o is the prevention of unauthorized disclosure of information.

o In other words, confidentiality means keeping information private or safe.
o it may be important for military, business or personal reasons.
o it may also be known as privacy or secrecy.

Computer Security module | PDF page 11 | lines 80-87

6. Access control
• Since many systems (such as router access control lists) define which packets may and which packets may not pass based on the sender's IP address
• An access enforcement mechanism authorizes requests from multiple subjects (e.g. users, processes, etc.) to perform operations (e.g., read, write, etc.) on objects (e.g., files, sockets, etc.).

Question sources

Computer Science MCQ practice collection · Question 563

Question 107 | CS0174

XYZ Company is attacked by intruders causing system crashes, slowing down of services and denying services to targeted users. The company is attacked by?

A. Access attack

B. Viruses

C. Information gathering

D. Denial of Services (DoS)

Hint

Match the resource bottleneck or wireless mechanism to the relevant protection layer.

Answer D: Denial of Services (DoS)

The decisive symptom is denial of service to intended users; that is the module's DoS definition (PDF pp.3,5). Crashes and slowing can implement that outcome without uniquely identifying malware.

Why each choice is right or wrong

A - Incorrect: Access attacks concern gaining control, not the specific denial outcome. See PDF p.3, PDF p.5.

B - Incorrect: A virus could be a cause, but no virus evidence is stated. See PDF p.3, PDF p.5.

C - Incorrect: Information gathering does not by itself deny users service. See PDF p.3, PDF p.5.

D - Correct: Denial of Services directly names the reported blocked access. See PDF p.3, PDF p.5.

Primary note proof

Study-note section SEC06

DoS exhausts or disables a resource; distributed DoS uses multiple sources, often bots or reflected/amplified responses. A SYN flood leaves many incomplete TCP handshakes occupying state. DNS flooding targets query capacity; NXDOMAIN attacks generate queries for nonexistent names to stress resolvers. Fragment floods abuse reassembly or packet-processing resources. Rate limits, SYN cookies, filtering and upstream capacity controls address different bottlenecks.

Study-note section SEC06

Rogue access points imitate or bypass approved wireless service.

Study-note section SEC04

Malware is the umbrella term, not a specific replication method.

Related reading (context, not primary note proof)

Study-note section SEC03

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 42-47

• Denial of Service (DoS) Attack: Blocking access from legitimate users
• Distributed DoS Attack: is accomplished by tricking routers into attacking a target or using Zombie hosts to simultaneously attack a given target with large number of packets.

Computer Security module | PDF page 3 | lines 87-91

• Availability:
o Assures that systems work promptly and service is not denied to authorized users.
o Information should be accessible and usable upon appropriate demand by an authorized user.

Question sources

Computer Science Exit Exam practice collection · Question 174

Question 108 | CS0175

Which one of the following is wrongly coupled about passive (offline) and active (online) attacks?

A. Active attack -> add or delete messages

B. Active attack -> traffic analysis

C. Passive attack -> monitoring traffic

D. Passive attack -> obtain message contents

Hint

Classify the action actually described and distinguish observation, alteration, disruption and identity claims.

Answer B: Active attack -> traffic analysis

PDF p.4 defines passive attacks as learning information without affecting resources and active attacks as alteration/effect on operation. Merely analyzing observed traffic fits passive learning, so B is the wrong pairing.

Why each choice is right or wrong

A - Incorrect: Adding/deleting messages changes resources and is active. See PDF p.4.

B - Correct: Traffic analysis, when only observing, is passive rather than active. See PDF p.4.

C - Incorrect: Monitoring without interference is passive. See PDF p.4.

D - Incorrect: Obtaining contents without changing resources is passive. See PDF p.4.

Primary note proof

Study-note section SEC03

> Interception copies or observes data, modification alters it, interruption blocks it, and fabrication inserts false material. The 4D classification uses disclosure, deception, disruption and usurpation, the last meaning unauthorized control. DoS primarily demonstrates disruption/availability loss; it need not give control of the victim.

Study-note section SEC03

> Passive attacks observe without changing the target exchange: sniffing and traffic analysis.

Study-note section SEC03

> These are not synonyms for offline and online.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 39-44

> • We can distinguish two types of attacks: (W. Stallings)
> o Active attack: an attempt to alter/affect system

resources or their operation.
o Passive attack: an attempt to learn/make use of information from the system that does not affect system resources.

Computer Security module | PDF page 4 | lines 50-61

• Types of Attacks on Data CIA (W. Stallings)
o Disclosure/Interception: unauthorized party snooping, or getting access to a resource, Attack on data confidentiality:
o Modification / Deception: unauthorized party modifying a resource, acceptance of false data, E.g., providing wrong data (attack on data integrity)
o Disruption/Interruption: destroyed/unavailable services/resources, DoS (attack on data availability)
o Fabrication: unauthorized party inserts a fake asset/resource, loss of confidentiality, authenticity and integrity of the message

Question sources

Computer Science Exit Exam practice collection · Question 175

Question 109 | CS0176

Which one of the following is NOT correct about symmetric key?

A. It requires sender and receiver to share a common secret key

B. Previous key cannot be used for a new key

C. Trusted third party can generate and deliver keys

D. Repetitive key change is encouraged to limit data being compromised

Hint

Distinguish shared-key architecture, key distribution and bulk-data use.

Answer B: Previous key cannot be used for a new key

PDF p.8 explicitly permits use of a previous key to encrypt a new key, so B contradicts a stated symmetric key-distribution alternative. A and C are explicit, while session-key disposal supports limiting reuse.

Why each choice is right or wrong

A - Incorrect: Sharing a common secret is explicitly required. See PDF p.7, PDF p.8.

B - Correct: B denies prior-key use that p.8 explicitly allows for distributing a new key. See PDF p.7, PDF p.8.

C - Incorrect: A third party can select and deliver keys to both participants. See PDF p.7, PDF p.8.

D - Incorrect: Discarding session keys after one session supports limiting key reuse and exposure. See PDF p.7, PDF p.8.

Primary note proof

Study-note section SEC16

Symmetric encryption uses a shared secret and efficiently protects bulk data. Pairwise independent keys for n users require n(n-1)/2 shared keys; a trusted key distribution center changes that architecture. A session key is temporary for a communication context; a master/key-encryption key protects subordinate keys. Distribute keys physically, via an authenticated protected channel, or through a trusted third party. An uncompromised old key can protect a new key, although this does not restore secrecy after the old key is stolen.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 7 | lines 48-54

Symmetric key cryptography
• Uses conventional / secret-key / single-key.
• The secret key shared by both sender and receiver.
• All classical encryption algorithms are private-key.
o Use the same key (shared key) for encryption and
decryption process.
• If this key is disclosed communications are compromised.

Computer Security module | PDF page 8 | lines 4-20

• Given parties A and B have various key distribution
alternatives:
o A can select key and physically deliver to B.
o Third party can select & deliver key to A & B.
o If A & B have communicated previously can use
previous key to encrypt a new key.
o If A & B have secure communications with a third-
party C, C can relay key between A & B.
• Key Hierarchy
o session key
✓ temporary key
✓ used for encryption of data between users
✓ for one logical session then discarded
o master key
✓ used to encrypt session keys
✓ shared by user & key distribution center
• Advantage: Simpler and faster

Question sources

Computer Science Exit Exam practice collection · Question 176

Question 110 | CS0177

Which one of the following is NOT a wireless attack?

A. Bluesnarfing

B. Phishing

C. Rogue access point

D. Bluejacking

Hint

Identify the trust claim or communication mapping being abused.

Answer B: Phishing is not specifically a wireless attack

SEC06 defines Bluetooth information theft as bluesnarfing, unsolicited Bluetooth messages as bluejacking and rogue access points as wireless service impersonation/bypass. Phishing is social deception across many media, not a wireless-specific mechanism. It can be delivered over a wireless connection, but that transport does not make its defining attack category wireless.

Why each choice is right or wrong

A - Incorrect: Bluesnarfing steals information over Bluetooth and is specifically a wireless attack mechanism.

B - Correct: Phishing is medium-independent trusted-source deception rather than a wireless-specific mechanism.

C - Incorrect: A rogue access point imitates or bypasses approved wireless service.

D - Incorrect: Bluejacking sends unsolicited Bluetooth messages, making it a wireless-specific attack category.

Primary note proof

Study-note section SEC05

> Phishing impersonates a trusted source to obtain secrets or induce an unsafe action; spear phishing targets a specific person. Verify a request through an independently known channel. Shoulder surfing observes entry; pretexting invents a reason for disclosure. Monitoring software used with authorization is not necessarily an attack merely because it records activity.

Study-note section SEC06

> Bluesnarfing steals Bluetooth information; bluejacking sends unsolicited Bluetooth messages.

Study-note section SEC06

> Rogue access points imitate or bypass approved wireless service.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 65-74

• Reconnaissance: The hackers first identifies a target to launch an attack, extract maximum information regarding this target, understand its vulnerabilities, and then only explore the best ways to exploit it.
o Examples: Credential Theft using Keyloggers and Spyware; Identity Theft using spam e-mail, phishing attack and social engineering attack
• Access: Ones the attacker Reconnaissance the target, S/he starts a successful exploitation and continued access to the system. Then, the attacker is in control of the target.

Question sources

Computer Science Exit Exam practice collection · Question 177

Question 111 | CS0178

Considering a vulnerability shared by identically configured components, which statement is NOT correct? Interpret diversity's benefits as possible reductions in common-mode compromise, not guarantees of minimum loss.

A. Non-uniform configurations may reduce loss from a vulnerability shared by one configuration

B. Uniform configurations can increase predictability

C. A vulnerability in the identical shared configuration cannot affect the other identically configured components

D. Diverse configurations may avoid exposing every resource to the same configuration-specific vulnerability

Hint

Start with the stated attacker path or physical hazard and the limits of each control.

Answer C: A vulnerability in the identical shared configuration cannot affect the other identically configured components

Uniform configurations simplify prediction and administration, but a vulnerability in the shared configuration can create correlated failures across components. Diversity may reduce that common-mode exposure, although it increases administrative cost and does not guarantee minimum loss. Option C wrongly denies the possibility of the same configuration vulnerability affecting peers. The other statements now describe qualified benefits rather than absolute guarantees.

Why each choice is right or wrong

A - Incorrect: Diversity can reduce exposure to a vulnerability limited to one configuration.

B - Incorrect: Uniformity can simplify prediction and administration.

C - Correct: Identical shared vulnerabilities can affect peer components, so this denial is false.

D - Incorrect: Diversity can reduce common-mode exposure without guaranteeing immunity.

Primary note proof

Study-note section SEC11

> Protect equipment against theft, unauthorized connections, heat, humidity, water, electrical surges and fire. A UPS supplies temporary conditioned power; it is normally protective, but faulty installation/battery failure can itself be hazardous. A disconnected machine loses many network attack paths yet remains exposed to insiders, removable media, supply-chain attacks and physical access. There is no universal “most secure computer” without a threat model.

Study-note section SEC11

> Uniform configurations simplify patching and prediction but create correlated failures; diversity may reduce common-mode compromise while raising administration cost.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 5 | lines 2-10

> • Principles of Least Privilege: —Do not give any more privileges than absolutely necessary to do/perform the required job‖.
> • Defense in Depth: having more than one layer or type of defense
> • Minimization: —do not run any software, or applications that are not strictly required to do the entrusted job.‖
> • Compartmentalization: limits the damage in other compartments when one compartment is compromised.

Question sources

Computer Science Exit Exam practice collection · Question 178

Question 112 | CS0184

_____ is a system that monitors traffic into and out of a network and automatically alerts personnel when suspicious traffic patterns occur.

A. Anti-virus Software

B. IDS

C. Firewall

D. Router

Hint

Separate observation from blocking and sensor placement from detection method.

Answer B: Intrusion detection system

Traffic observation followed by alerts about suspicious patterns is the IDS function in SEC24. The stem does not require inline blocking. Antivirus, firewalls and routers have different primary functions, though products can combine capabilities. B identifies the functional system described without assuming a firewall always detects every intrusion.

Why each choice is right or wrong

A - Incorrect: Antivirus targets malicious software and is not the generic traffic-pattern alerting system named in the note.

B - Correct: An IDS observes traffic/activity and alerts personnel to suspected intrusion patterns.

C - Incorrect: A firewall primarily enforces traffic policy; observation-and-alerting specifically identifies intrusion detection.

D - Incorrect: Routing selects packet next hops rather than defining suspicious-pattern detection and alerts.

Primary note proof

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline. Host-based sensors examine host events, network-based sensors inspect traffic. Signature detection recognizes known patterns; anomaly detection compares with a baseline and may detect novel behavior. Neither guarantees every attack is detected; anomaly detection can raise many false positives, and signatures can sometimes match variants.

Study-note section SEC23

> A firewall enforces traffic policy.

Study-note section SEC-S03

> A router forwards IP packets between networks; forwarding alone is different from detecting suspicious behavior or enforcing a firewall's security policy.

Related reading (context, not primary note proof)

Study-note section SEC04

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Computer Security module | PDF page 4 | lines 89-99

> o Encryption: primary control that protects CIA, by
> which Cleartext scambled into ciphertext.
> o Software controls: password checker, virus scanner,
> IDS (intrusion detection system)
> o Hardware controls: provide higher degree of security;
> Locks, Smart cards, dongles, hadware keys...
> o Policies and procedures: Alignment with users' legal

and ethical standards; eg. password policy
o Physical controls: natural and man-made disaster protection; walls, locks, guards, security, cameras, backup copies and archives

Computer Security module | PDF page 4 | lines 82-84

o Detect attack: accept that an attack will occur; the goal is to determine that an attack is under way, or has occurred, and report it.

Computer Security module | PDF page 11 | lines 22-34

3. Firewall
• a program or network devices (e.g. router) which filters access to a protected network from the internet connection.
• protects a local or network of systems from network-based security threats,
• monitors & controls incoming & outgoing network traffic based on predetermined security rules.
• Firewall Objectives
o Keep intruders, malicious code and unwanted traffic out
o Keep private and sensitive information in
o security wall b/n private (protected) NW & outside word.

Question sources

Computer Science Exit Exam practice collection · Question 184

Question 113 | NS0327

Suppose Mr. Negesa might act as Mr. Dereje and send message to Registrar X, the registrar might be lead to believe that message indeed come from Mr. Dereje. What types of attack Mr. Negesa has committed?

A. Denial of service

B. Modification of message

C. Masquerade

D. Replay

Hint

Identify the trust claim or communication mapping being abused.

Answer C: Masquerade

Negesa presents the message as coming from Dereje, so the registrar attributes it to the wrong person. SEC05 defines that false identity claim as masquerade. No service exhaustion, alteration of a prior message or reuse of an old valid message is stated; those attack mechanisms must not be inferred from impersonation alone.

Why each choice is right or wrong

A - Incorrect: The scenario does not deny service or exhaust resources.

B - Incorrect: No prior message is modified; the central behavior is a false sender identity.

C - Correct: Claiming Dereje's identity to fool the registrar is masquerade.

D - Incorrect: Replay requires reusing a valid old message, which the scenario does not describe.

Primary note proof

Study-note section SEC05

> Masquerade claims another identity. ARP spoofing poisons local IPv4-to-MAC associations; DNS spoofing/cache poisoning supplies false name resolutions; IP spoofing falsifies a source address. An intermediary attack intercepts and can alter exchanges. A replay reuses a valid old message; nonces, timestamps and sequence checks can reject it. Session hijacking steals or predicts a valid session token. Protect transport, regenerate session IDs after login, restrict cookies and check authorization server-side on every sensitive request.

Study-note section SEC06

> Rogue access points imitate or bypass approved wireless service.

Study-note section SEC06

> Fragment floods abuse reassembly or packet-processing resources.

Related reading (context, not primary note proof)

Study-note section SEC03

Question sources

Computer Science Exit Exam · 2018 · Question 54

Question 114 | NS0343

Which one of following monitors incoming and outgoing network traffic and decides whether to allow or block specific traffic based on a defined set of security rules?

A. Intrusion detection system (IDS)

B. Proxy server

C. Virtual Private network

D. Firewall

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer D: Firewall

The described allow/block decisions follow a defined traffic policy, the firewall function in SEC23. IDS observes/alerts, a proxy mediates application requests and a VPN supplies logical private connectivity. A proxy may implement policy in some deployments, but the generic boundary traffic-policy device directly named by the definition is a firewall.

Why each choice is right or wrong

A - Incorrect: An IDS observes and alerts rather than being defined by enforcing allow/block rules.

B - Incorrect: A proxy mediates application exchanges; the general traffic-policy enforcement definition specifically names a firewall.

C - Incorrect: A VPN extends private connectivity and is not inherently a generic rule-based traffic filter.

D - Correct: A firewall enforces allow/block traffic policy based on configured rules.

Primary note proof

Study-note section SEC23

> A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Study-note section SEC22

> A VPN is a logical private connection; its actual security depends on the chosen authenticated tunnel.

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline.

Question sources

Computer Science Exit Exam · 2018 · Question 76

Question 115 | NS0361

What is the difference between confidentiality, integrity, and availability? Give an example of a breach of each.

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer: Confidentiality limits disclosure: copying a private credit-card database breaches it. Integrity preserves correctness: changing an account balance improperly breaches it. Availability provides timely usable service: crashing a server or starving resources breaches it.

SEC01 defines the three goals by the property harmed, not by one fixed attack label. Copying can breach secrecy without changing the original; alteration harms correctness; denied service harms availability. One incident can violate several goals, so these examples illustrate the primary property rather than mutually exclusive incident classes.

Primary note proof

Study-note section SEC01

> Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Question sources

Computer Security review questions · Question 1

Question 116 | NS0362

Why is multi-factor authentication more secure than passwords alone?

Hint

Separate identity claims from proof, and factor type from factor count.

Answer: MFA requires independent factor classes, such as a password plus a possessed card or biometric. Stealing the password alone then does not satisfy the other proof. Password plus PIN is still knowledge-only, not two independent factors.

SEC07 distinguishes knowledge, possession and inherence. Combining classes adds a separate proof rather than merely another secret from the same class. That reduces dependence on a password alone, although implementation, recovery channels and compromise of the additional factor still matter; MFA is not a guarantee against every attack.

Primary note proof

Study-note section SEC07

> Knowledge is a password/PIN, possession is a token/card, and inherence is a biometric such as a fingerprint. Behavioral biometrics include typing rhythm or handwriting; some textbooks call these “something one does.” A body characteristic is not “three-factor authentication”: one biometric is one factor. Password plus PIN remains one factor class; card plus PIN is two. An OTP’s factor depends on its delivery/generation mechanism.

Study-note section SEC08

> Use unique long secrets and password managers, rate limiting and suitable MFA; store salted slow password hashes, not reversible plaintext passwords.

Study-note section SEC08

> OAuth alone does not define a complete end-user login identity protocol; OpenID Connect adds an identity layer.

Question sources

Computer Security review questions · Question 2

Question 117 | NS0363

What is the difference between symmetric and asymmetric encryption? Why do we use both?

Hint

Identify the intended recipient or signer and separate confidentiality from verification.

Answer: Symmetric encryption shares a secret key and efficiently protects bulk data. Asymmetric encryption uses a recipient public/private pair: encrypt to the authenticated public key and decrypt with the private key. Hybrid systems use public-key mechanisms to establish a symmetric secret, then symmetric protection for content.

SEC16 supplies the efficient shared-secret model; SEC18 supplies recipient key direction and hybrid composition. Public keys ease distribution only with authentic identity bindings. Combining them addresses distribution/authentication and bulk-data efficiency without claiming either family is inherently more secure or assigning fixed speed ratios.

Primary note proof

Study-note section SEC18

> For confidentiality to B, encrypt with B's authenticated public encryption key; B decrypts with B's private key. Do not encrypt to A's public key if only B should read it, and never assume A possesses B's private key. For a signature, A signs with A's private signing key and others verify with A's public key. These are distinct operations; not every public-key algorithm supports encryption.

Study-note section SEC16

> Symmetric encryption uses a shared secret and efficiently protects bulk data.

Study-note section SEC18

> Hybrid systems establish or encapsulate a symmetric secret with public-key mechanisms, then protect bulk data symmetrically.

Question sources

Computer Security review questions · Question 3

Question 118 | NS0364

What is a DDoS attack and how does it differ from a DoS attack?

Hint

Match the resource bottleneck or wireless mechanism to the relevant protection layer.

Answer: DoS exhausts or disables a resource so authorized users lose usable service. DDoS is DoS launched through multiple distributed sources, often bots or reflected/amplified responses. The distinction is source distribution, not a different security goal: both primarily harm availability.

SEC06 defines the resource-exhaustion/disabling mechanism and distributed sources. Multiple sources can stress different bottlenecks, so appropriate responses include rate limits, filtering and upstream capacity controls. The number of sources does not imply the attacker obtained control of the victim; the demonstrated harm is unavailable service.

Primary note proof

Study-note section SEC06

> DoS exhausts or disables a resource; distributed DoS uses multiple sources, often bots or reflected/amplified responses. A SYN flood leaves many incomplete TCP handshakes occupying state. DNS flooding targets query capacity; NXDOMAIN attacks generate queries for nonexistent names to stress resolvers. Fragment floods abuse reassembly or packet-processing resources. Rate limits, SYN cookies, filtering and upstream capacity controls address different bottlenecks.

Study-note section SEC03

> DoS primarily demonstrates disruption/availability loss; it need not give control of the victim.

Study-note section SEC06

> Rogue access points imitate or bypass approved wireless service.

Question sources

Computer Security review questions · Question 4

Question 119 | NS0365

Why is SQL injection dangerous? How is it prevented?

Hint

Locate where untrusted data can become instructions and where they execute.

Answer: SQL injection lets untrusted input alter SQL syntax through unsafe string construction, changing query behavior or affected rows. Use prepared statements with bound data values, allow-list nonparameterizable identifiers, and give the application account minimal privileges. Stored procedures/ORMs must also avoid unsafe dynamic SQL.

The danger is crossing the data-versus-query-syntax boundary, not merely accepting an unusual string. Parameter binding keeps supplied values as data, while identifier allow-lists handle structure that cannot be bound. Restricting database rights limits impact. SEC26 warns that a procedure or ORM name alone does not make dynamically concatenated SQL safe.

Primary note proof

Study-note section SEC26

> SQL injection occurs when untrusted values alter SQL syntax through unsafe string construction. A tautology can change which rows match. Bind data values in prepared statements, allow-list identifiers that cannot be parameters, and restrict the application's database account. Stored procedures/ORMs are safe only when they avoid unsafe dynamic SQL.

Study-note section SEC26

> Stored content persists; reflected content returns in a response; DOM-based XSS can arise from unsafe browser sinks.

Question sources

Computer Security review questions · Question 5

Question 120 | NS0366

What is the difference between IDS and IPS?

Hint

Separate observation from blocking and sensor placement from detection method.

Answer: An IDS observes activity and alerts about suspected attacks. An IPS can actively block traffic, usually inline. Detection can use signatures or anomaly baselines; neither system guarantees every attack is detected or every alert represents a real attack.

The key distinction in SEC24 is observation/alerting versus enforcement. Host sensors examine host events and network sensors inspect traffic, independently of that IDS/IPS action distinction. Inline prevention adds a blocking decision; false positives can then deny legitimate activity, so detection context and policy remain important.

Primary note proof

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline. Host-based sensors examine host events, network-based sensors inspect traffic. Signature detection recognizes known patterns; anomaly detection compares with a baseline and may detect novel behavior. Neither guarantees every attack is detected; anomaly detection can raise many false positives, and signatures can sometimes match variants.

Study-note section SEC24

> A true positive is a real attack detected, false negative is a missed attack, false positive is innocent activity flagged, and true negative is innocent activity accepted.

Question sources

Computer Security review questions · Question 6

Question 121 | NS0417

What are the three primary security goals known as the CIA triad?

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer: The CIA triad is confidentiality, integrity and availability: restrict unauthorized disclosure, prevent improper changes, and provide timely usable service to authorized users.

SEC01 defines each goal by a separate property. Secrecy concerns who learns information, integrity concerns correct information/changes, and availability concerns usable service. These goals can be harmed together; satisfying one does not establish the others. Authentication and authorization support protection but are not replacements for the three CIA goals.

Primary note proof

Study-note section SEC01

> Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC01

> Authentication establishes an identity or other claim; authorization decides permitted actions.

Study-note section SEC01

> Data security protects information at rest, in use and in transit, including accidental corruption as well as intentional attacks.

Question sources

Computer Security review questions · Question 1

Question 122 | NS0418

What is a 'vulnerability' in the context of computer security?

Hint

Distinguish weaknesses, potential harmful events, actual attempts and the effect of a control.

Answer: A vulnerability is an exploitable weakness in a system or its protection. It differs from a threat, which is a potentially harmful circumstance, and an attack, which is an actual attempt to cause harm.

SEC02 separates weakness, potential harm and attempted exploitation. A software flaw can exist without an attack taking place; controls reduce likelihood or impact. Calling every flaw an active attack would erase that distinction and misstate what a vulnerability finding actually proves.

Primary note proof

Study-note section SEC02

> An asset is something valuable: data, software, hardware, service or reputation. A threat is a potentially harmful circumstance; a vulnerability is an exploitable weakness; an attack is an actual attempt; a control reduces likelihood or impact. A software flaw is not automatically an active attack. Risk concerns likelihood and consequence, not a universally meaningful numeric product of two unmeasured labels.

Question sources

Computer Security review questions · Question 2

Question 123 | NS0419

What is the difference between 'prevention' and 'detection' in computer security?

Hint

Distinguish weaknesses, potential harmful events, actual attempts and the effect of a control.

Answer: Prevention attempts to block a harmful event; detection reveals that an event or suspicious activity has occurred. Encryption can prevent disclosure, while audit logs supply detective/accountability evidence. Recovery is a further function that restores safe operation after damage.

SEC02 classifies controls by effect rather than by whether they are physical or technical. A preventive control can fail, and a detective control may then expose the event. Detection does not itself imply blocking; combining these stages supports a fuller protection and response process.

Primary note proof

Study-note section SEC02

> Preventive controls block events, detective controls reveal them, corrective/recovery controls restore safe operation, and deterrents discourage attempts. Encryption is preventive for disclosure; an audit log is principally detective/accountability evidence. Physical, technical and administrative are control categories, not competing stages. An external firewall cannot replace a written policy or a locked equipment room.

Question sources

Computer Security review questions · Question 3

Question 124 | NS0420

Which security property ensures that a message was not altered by unauthorized individuals?

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer: Integrity is the property that protects a message against unauthorized or improper alteration. A suitable keyed MAC or digital signature can provide integrity evidence under the relevant key/trust assumptions.

SEC01 defines integrity by correctness of content and changes. SEC20 explains that an unkeyed hash alone requires a trusted reference because an attacker can recompute it after modification. Therefore the property is integrity, while the mechanism used to establish it must be selected with an appropriate trust model.

Primary note proof

Study-note section SEC01

> Confidentiality restricts disclosure: copying a credit-card database violates secrecy even if the original stays intact. Integrity restricts unauthorized or improper changes: altering a balance or inserting a false message damages correctness. Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it. One event can harm several goals.

Study-note section SEC20

> An attacker can recompute an unkeyed hash of modified data, so a hash alone needs a trusted reference.

Study-note section SEC20

> Verification checks origin/integrity under key-identity and key-custody assumptions; it does not conceal content.

Question sources

Computer Security review questions · Question 4

Question 125 | NS0421

What is Data Security focused on protecting?

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer: Data security protects information at rest, in use and in transit against unauthorized disclosure, improper changes and loss of usable access. It includes accidental corruption as well as deliberate attacks.

SEC01 explicitly covers the information lifecycle and accidental as well as intentional harm. Protection is broader than encrypting network traffic: storage, processing and access/recovery controls also matter. The CIA goals describe complementary properties, so secrecy alone is not the whole of data security.

Primary note proof

Study-note section SEC01

> Authentication establishes an identity or other claim; authorization decides permitted actions. Accountability links actions to responsible entities through reliable identity and protected logs. Non-repudiation supplies evidence supporting origin or receipt; signatures do not make denial physically impossible. Information assurance includes confidence, risk management, protection, detection, response and restoration across the information lifecycle. Data security protects information at rest, in use and in transit, including accidental corruption as well as intentional attacks.

Question sources

Computer Security review questions · Question 5

Question 126 | NS0422

What are the '4D's' of security threats?

Hint

Classify the action actually described and distinguish observation, alteration, disruption and identity claims.

Answer: The four are disclosure, deception, disruption and usurpation. Disclosure exposes information; deception supplies false material or beliefs; disruption harms service; usurpation means unauthorized control.

SEC03 explicitly names this classification and distinguishes disruption from control. A denial-of-service event demonstrates disruption and availability loss, but does not by itself prove the attacker usurped the victim. The categories describe kinds of harm rather than requiring every attack to produce all four.

Primary note proof

Study-note section SEC03

> Interception copies or observes data, modification alters it, interruption blocks it, and fabrication inserts false material. The 4D classification uses disclosure, deception, disruption and usurpation, the last meaning unauthorized control. DoS primarily demonstrates disruption/availability loss; it need not give control of the victim.

Question sources

Computer Security review questions · Question 6

Question 127 | NS0423

What is the key difference between passive and active attacks?

Hint

Classify the action actually described and distinguish observation, alteration, disruption and identity claims.

Answer: Passive attacks observe without changing the exchange, such as sniffing or traffic analysis. Active attacks inject, modify, replay or disrupt communications or service. Passive/active does not mean offline/online.

The distinction is intervention in the target exchange, as SEC03 explains. Traffic analysis can infer relationships from timing and sizes even with encrypted content, so passive observation still creates risk. Neither category is universally easy to detect or prevent; those claims depend on the mechanism and deployed controls.

Primary note proof

Study-note section SEC03

> Interception copies or observes data, modification alters it, interruption blocks it, and fabrication inserts false material. The 4D classification uses disclosure, deception, disruption and usurpation, the last meaning unauthorized control. DoS primarily demonstrates disruption/availability loss; it need not give control of the victim.

Study-note section SEC03

> Passive attacks observe without changing the target exchange: sniffing and traffic analysis.

Study-note section SEC03

> Traffic analysis can infer relationships from timing/size even when content is encrypted.

Question sources

Computer Security review questions · Question 7

Question 128 | NS0424

What is a Denial of Service (DoS) attack?

Hint

Match the resource bottleneck or wireless mechanism to the relevant protection layer.

Answer: A denial-of-service attack exhausts or disables a resource so authorized users cannot obtain timely usable service. Examples include incomplete-handshake state exhaustion or resolver query flooding. Distributed DoS uses multiple sources.

SEC06 identifies the mechanism as service exhaustion/disabling. SEC01 connects unavailable or unusable service to availability loss. The attacker need not alter stored data or gain control of the target; the resource bottleneck and service effect distinguish this attack from other harms.

Primary note proof

Study-note section SEC06

> DoS exhausts or disables a resource; distributed DoS uses multiple sources, often bots or reflected/amplified responses. A SYN flood leaves many incomplete TCP handshakes occupying state. DNS flooding targets query capacity; NXDOMAIN attacks generate queries for nonexistent names to stress resolvers. Fragment floods abuse reassembly or packet-processing resources. Rate limits, SYN cookies, filtering and upstream capacity controls address different bottlenecks.

Study-note section SEC01

> Availability supplies timely, usable service to authorized users: crashes, resource starvation and denied access violate it.

Study-note section SEC06

> Rogue access points imitate or bypass approved wireless service.

Question sources

Computer Security review questions · Question 8

Question 129 | NS0425

What does 'snooping' refer to in network security?

Hint

Classify the action actually described and distinguish observation, alteration, disruption and identity claims.

Answer: Snooping is unauthorized observation or interception of information, such as reading captured network traffic. When it only observes the exchange without modifying it, it is a passive attack against confidentiality.

SEC03 groups interception with observing/copying data and lists sniffing among passive attacks. This differs from modification or fabrication, which change or inject material. Encryption can hide content while traffic timing/size still supports traffic analysis, so observing communication metadata can remain informative.

Primary note proof

Study-note section SEC03

> Interception copies or observes data, modification alters it, interruption blocks it, and fabrication inserts false material. The 4D classification uses disclosure, deception, disruption and usurpation, the last meaning unauthorized control. DoS primarily demonstrates disruption/availability loss; it need not give control of the victim.

Study-note section SEC03

> Passive attacks observe without changing the target exchange: sniffing and traffic analysis.

Study-note section SEC03

> Traffic analysis can infer relationships from timing/size even when content is encrypted.

Question sources

Computer Security review questions · Question 9

Question 130 | NS0426

What is 'repudiation' in computer security?

Hint

Identify the property harmed: disclosure, improper change, unavailable service or lack of reliable attribution.

Answer: Repudiation is denying involvement, such as denying having sent or received a message. Non-repudiation supplies evidence supporting origin or receipt under identity/key-custody assumptions; it does not make a person physically unable to deny an action.

SEC01 frames non-repudiation as evidentiary support rather than an absolute ban on denial. SEC20 explains why signatures can support origin/integrity while shared-key MACs limit third-party attribution because both holders can create them. The strength of the evidence depends on trustworthy identities, keys and records.

Primary note proof

Study-note section SEC01

> Authentication establishes an identity or other claim; authorization decides permitted actions. Accountability links actions to responsible entities through reliable identity and protected logs. Non-repudiation supplies evidence supporting origin or receipt; signatures do not make denial physically impossible. Information assurance includes confidence, risk management, protection, detection, response and restoration across the information lifecycle. Data security protects information at rest, in use and in transit, including accidental corruption as well as intentional attacks.

Study-note section SEC20

> A keyed MAC provides integrity and shared-secret authentication; both holders can create it, limiting third-party non-repudiation.

Study-note section SEC20

> Verification checks origin/integrity under key-identity and key-custody assumptions; it does not conceal content.

Question sources

Computer Security review questions · Question 10

Question 131 | NS0427

What distinguishes a worm from a computer virus?

Hint

Separate propagation, disguise and payload when classifying malicious software.

Answer: A virus attaches to a host program/document and spreads when the infected host executes. A worm replicates independently, commonly across networks. The distinction is host dependency, not a universal ranking of speed or required user action.

SEC04 defines propagation mechanisms separately from payloads and disguise. A malware package may contain several components, so a worm can also carry another malicious function. That does not change the independent-replication distinction or justify assuming every worm always spreads faster than every virus.

Primary note proof

Study-note section SEC04

> A virus replicates by attaching to a host program or document; execution of the infected host spreads it. A worm replicates independently, commonly across networks. A Trojan disguises malicious behavior as something useful and does not require self-replication. These labels describe different properties, so a malware package may contain several components.

Study-note section SEC04

> Adware displays unwanted advertising and may also track users.

Study-note section SEC04

> Malware is the umbrella term, not a specific replication method.

Question sources

Computer Security review questions · Question 11

Question 132 | NS0428

What is a Trojan Horse in cybersecurity?

Hint

Separate propagation, disguise and payload when classifying malicious software.

Answer: A Trojan horse presents apparently useful or legitimate behavior while hiding malicious activity. Unlike virus/worm labels, the definition concerns disguise and does not require self-replication.

SEC04 distinguishes propagation, disguise and payload properties. A Trojan may contain other malware components, but merely looking useful while concealing harmful behavior is the identifying clue. It should not be defined as a worm simply because its payload might later spread.

Primary note proof

Study-note section SEC04

> A virus replicates by attaching to a host program or document; execution of the infected host spreads it. A worm replicates independently, commonly across networks. A Trojan disguises malicious behavior as something useful and does not require self-replication. These labels describe different properties, so a malware package may contain several components.

Study-note section SEC04

> A backdoor bypasses ordinary access paths; a rootkit hides malicious presence or privileged activity.

Study-note section SEC04

> Malware is the umbrella term, not a specific replication method.

Question sources

Computer Security review questions · Question 12

Question 133 | NS0429

What is a Logic Bomb?

Hint

Separate propagation, disguise and payload when classifying malicious software.

Answer: A logic bomb is malicious code that waits for a specified trigger before activating its payload. The defining feature is conditional activation, rather than a particular replication method or inevitable immediate execution.

SEC04 identifies the trigger dependency. Other malware terms describe independent replication, host attachment, disguise or hidden access; those properties can coexist with a triggered payload. Thus the answer identifies when the harmful behavior activates without inventing a particular date, event or payload not supplied by the question.

Primary note proof

Study-note section SEC04

> A virus replicates by attaching to a host program or document; execution of the infected host spreads it. A worm replicates independently, commonly across networks. A Trojan disguises malicious behavior as something useful and does not require self-replication. These labels describe different properties, so a malware package may contain several components.

Study-note section SEC04

> A logic bomb waits for a trigger.

Study-note section SEC04

> Malware is the umbrella term, not a specific replication method.

Question sources

Computer Security review questions · Question 13

Question 134 | NS0430

What is the difference between a White Hat and Black Hat hacker?

Hint

Classify the action actually described and distinguish observation, alteration, disruption and identity claims.

Answer: White-hat security testing is authorized and stays within agreed scope. Black-hat activity is malicious or unauthorized. Technical skill or the word hacker alone does not establish legality.

SEC03 makes authorization and scope the decisive distinction. Good claimed intentions do not create permission to test another system, and insider/outsider status describes a different dimension. An authorized tester must remain within the approved activity; the label alone cannot excuse actions outside that scope.

Primary note proof

Study-note section SEC03

> White-hat testing requires authorization and scope; black-hat activity is malicious or unauthorized. A competitor may seek commercial secrets; hacktivists pursue a cause. “Hacker” alone does not establish legality. Insider versus outsider describes origin/trust position, not the specific mechanism.

Study-note section SEC27

> In a check-then-open file operation, another actor may replace the file between the check and use.

Study-note section SEC27

> Correct findings, patch, test and recheck; testing is not proof of absence.

Question sources

Computer Security review questions · Question 14

Question 135 | NS0431

What is phishing?

Hint

Identify the trust claim or communication mapping being abused.

Answer: Phishing impersonates a trusted source to trick someone into revealing secrets or performing an unsafe action. Spear phishing targets a particular person. Verify suspicious requests using an independently known channel.

SEC05 defines the deception mechanism and the targeted variant. It is a human-trust attack, not necessarily a wireless mechanism or direct password-guessing algorithm. Independent verification matters because replying through the attacker-supplied contact path can preserve the impersonation rather than test it.

Primary note proof

Study-note section SEC05

> Phishing impersonates a trusted source to obtain secrets or induce an unsafe action; spear phishing targets a specific person. Verify a request through an independently known channel. Shoulder surfing observes entry; pretexting invents a reason for disclosure. Monitoring software used with authorization is not necessarily an attack merely because it records activity.

Question sources

Computer Security review questions · Question 15

Question 136 | NS0432

What is the primary goal of cryptography?

Hint

Trace message transformations and separate data, keys, algorithms and codebreaking.

Answer: Cryptography protects information using key-based mathematical mechanisms. Encryption primarily conceals meaning; suitable MACs/signatures or authenticated encryption additionally support integrity and authenticity. There is no single confidentiality-only description that captures every cryptographic service.

SEC12 defines encryption/decryption, while SEC20 distinguishes hashes, MACs and signatures. These mechanisms address different goals and assumptions: a signature does not conceal content, and an unkeyed hash alone does not establish authentic origin. The goal must be matched to the mechanism rather than treating all cryptography as encryption.

Primary note proof

Study-note section SEC12

> Plaintext is the original message, ciphertext its encrypted representation, a key is input to an algorithm, encryption computes C=E_k(P), and decryption recovers P=D_k(C). The five conventional symmetric ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. Cryptanalysis seeks recovery or other breaks without the authorized secret; cryptology includes cryptography and cryptanalysis. Steganography hides that a message exists; encryption hides its meaning.

Study-note section SEC20

> Verification checks origin/integrity under key-identity and key-custody assumptions; it does not conceal content.

Study-note section SEC20

> An attacker can recompute an unkeyed hash of modified data, so a hash alone needs a trusted reference.

Related reading (context, not primary note proof)

Study-note section SEC16

Question sources

Computer Security review questions · Question 16

Question 137 | NS0433

What is the difference between symmetric and asymmetric key cryptography?

Hint

Identify the intended recipient or signer and separate confidentiality from verification.

Answer: Symmetric cryptography uses a shared secret between parties. Asymmetric confidentiality uses an authenticated public encryption key and corresponding private decryption key; signatures instead use a private signing key and public verification key. Not every public-key algorithm supports encryption.

SEC16 gives the shared-secret model and SEC18 the asymmetric key directions. Symmetric keys require secure distribution and can efficiently protect bulk data. Public keys simplify distribution only when their identity bindings are authentic; hybrid protocols combine these advantages instead of assuming one family is inherently stronger.

Primary note proof

Study-note section SEC18

> For confidentiality to B, encrypt with B's authenticated public encryption key; B decrypts with B's private key. Do not encrypt to A's public key if only B should read it, and never assume A possesses B's private key. For a signature, A signs with A's private signing key and others verify with A's public key. These are distinct operations; not every public-key algorithm supports encryption.

Study-note section SEC16

> Symmetric encryption uses a shared secret and efficiently protects bulk data.

Study-note section SEC18

> Public keys ease distribution but require authentic bindings to identities.

Related reading (context, not primary note proof)

Study-note section SEC12

Question sources

Computer Security review questions · Question 17

Question 138 | NS0434

In Caesar's cipher with a shift of 3, what does the letter 'A' become?

Hint

Separate symbol replacement from position changes and apply the alphabet and modular direction.

Answer: A becomes D. With A=0 and shift k=3, C=(0+3) mod26=3, which is D.

SEC13 specifies the A=0 through Z=25 convention and forward addition for Caesar encryption. Applying that convention avoids confusing encryption with the reverse decryption shift or using a different numeric alphabet. No wraparound is needed for A, but the modulo rule handles letters near the alphabet end.

Primary note proof

Study-note section SEC13

> With A=0 through Z=25, Caesar encryption is C=(P+k) mod26 and decryption P=(C-k) mod26. A shift of 3 maps A→D and HELLO→KHOOR. Key25 moves back one: ATTACK→ZSSZBJ. Ensure negative results wrap into 0-25. A multiplicative cipher uses C=kP mod26 with gcd(k,26)=1; decryption uses k's modular inverse. Affine uses C=aP+b mod26 and requires gcd(a,26)=1. The inverse-key relation, not simply the same numeric multiplication, recovers plaintext.

Question sources

Computer Security review questions · Question 18

Question 139 | NS0435

What formula is used for encryption in the Caesar cipher?

Hint

Separate symbol replacement from position changes and apply the alphabet and modular direction.

Answer: Using A=0 through Z=25, Caesar encryption is C=(P+k) mod26. Here P is the plaintext letter number, k the shift and C the ciphertext letter number. Decryption subtracts the same shift modulo 26.

SEC13 fixes both the numeric alphabet and the direction. Modulo 26 wraps results into the letter range, including negative intermediate values in decryption. A stated shift of 3 therefore maps A to D; reversing the sign would describe decryption instead of the requested encryption formula.

Primary note proof

Study-note section SEC13

> With A=0 through Z=25, Caesar encryption is C=(P+k) mod26 and decryption P=(C-k) mod26. A shift of 3 maps A→D and HELLO→KHOOR. Key25 moves back one: ATTACK→ZSSZBJ. Ensure negative results wrap into 0-25. A multiplicative cipher uses C=kP mod26 with gcd(k,26)=1; decryption uses k's modular inverse. Affine uses C=aP+b mod26 and requires gcd(a,26)=1. The inverse-key relation, not simply the same numeric multiplication, recovers plaintext.

Question sources

Computer Security review questions · Question 19

Question 140 | NS0436

Why is the One-Time Pad considered the most secure cipher?

Hint

Check the precise ordering or key-use assumptions rather than relying on a cipher's name.

Answer: A genuine one-time pad has perfect secrecy when its key is independent, uniformly random, secret, at least as long as the message and never reused. Without every condition, a Vernam/XOR construction is not automatically a one-time pad.

SEC15 makes the guarantee conditional, not a blanket ranking of every cipher implementation. Pad reuse exposes C1 XOR C2=P1 XOR P2 and leaks relationships between plaintexts. Key delivery and storage are substantial practical costs even though the ideal construction supplies perfect secrecy under its assumptions.

Primary note proof

Study-note section SEC15

> A one-time pad XORs bits with an independent uniformly random secret key at least as long as the message and never reused. Its perfect secrecy depends on every condition. Reusing a pad leaks C1 XOR C2=P1 XOR P2. A Vernam XOR construction is not automatically a one-time pad: a predictable/reused keystream loses that guarantee. Key delivery and storage remain substantial costs.

Question sources

Computer Security review questions · Question 20

Question 141 | NS0437

What is the main weakness of a monoalphabetic substitution cipher?

Hint

Separate symbol replacement from position changes and apply the alphabet and modular direction.

Answer: One fixed substitution throughout a message preserves letter-frequency patterns, enabling statistical analysis despite a large nominal key space. This is the main weakness of monoalphabetic substitution.

SEC13 distinguishes symbol replacement from transposition and stresses that fixed mappings leak patterns. Increasing the nominal number of possible substitution alphabets does not erase language statistics. Classical substitution therefore serves teaching purposes rather than providing modern data protection, even if exhaustive enumeration of every alphabet is impractical.

Primary note proof

Study-note section SEC13

> Substitution replaces symbols; transposition changes their positions. A monoalphabetic substitution preserves one mapping throughout the message and leaks frequency patterns despite a large nominal key space. Classical ciphers are teaching devices, not modern data protection.

Question sources

Computer Security review questions · Question 21

Question 142 | NS0438

How does the Vigenere cipher improve on monoalphabetic ciphers?

Hint

Identify the cipher's unit, key structure and assumptions before judging its strength.

Answer: Vigenère applies position-dependent shifts from a key, so the same plaintext letter can encrypt differently at different positions. That reduces the single fixed-substitution frequency pattern, but repeating a short key still leaks periodic structure.

SEC14 gives C_i=(P_i+K_i) mod26 and explains the position dependence. The improvement is multiple mappings rather than a guarantee of modern security. A periodic key allows related positions to share the same shift, preserving exploitable regularities instead of eliminating statistical structure altogether.

Primary note proof

Study-note section SEC14

> Vigenère repeats or otherwise supplies shifts K_i and computes C_i=(P_i+K_i) mod26. The same plaintext letter may encrypt differently by position; repeated short keys still leak periodic structure. An autokey scheme extends its initial key using message material.

Question sources

Computer Security review questions · Question 22

Question 143 | NS0439

What is steganography and how does it differ from cryptography?

Hint

Trace message transformations and separate data, keys, algorithms and codebreaking.

Answer: Steganography conceals the existence of a message by hiding it within other data. Encryption conceals its meaning by transforming plaintext into ciphertext. Cryptography also includes integrity/authenticity mechanisms; it is broader than just encryption.

SEC12 states the existence-versus-meaning distinction. Discovering a hidden carrier can defeat concealment of existence without necessarily revealing separately encrypted content. SEC20's MAC/signature roles further explain why cryptography cannot be reduced to steganography or to secrecy of the communication's existence.

Primary note proof

Study-note section SEC12

> Plaintext is the original message, ciphertext its encrypted representation, a key is input to an algorithm, encryption computes C=E_k(P), and decryption recovers P=D_k(C). The five conventional symmetric ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. Cryptanalysis seeks recovery or other breaks without the authorized secret; cryptology includes cryptography and cryptanalysis. Steganography hides that a message exists; encryption hides its meaning.

Study-note section SEC20

> Verification checks origin/integrity under key-identity and key-custody assumptions; it does not conceal content.

Study-note section SEC20

> A keyed MAC provides integrity and shared-secret authentication; both holders can create it, limiting third-party non-repudiation.

Question sources

Computer Security review questions · Question 23

Question 144 | NS0440

What are the five ingredients of cryptography?

Hint

Trace message transformations and separate data, keys, algorithms and codebreaking.

Answer: For the conventional symmetric-encryption model, the five ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. This is not a universal inventory of every cryptographic construction.

SEC12 supplies this five-part teaching model. Plaintext enters encryption with a secret key, producing ciphertext; decryption recovers the plaintext with the appropriate secret. The qualifier symmetric matters because signature, hash and public-key constructions have different interfaces and cannot all be reduced to this particular five-item list.

Primary note proof

Study-note section SEC12

> Plaintext is the original message, ciphertext its encrypted representation, a key is input to an algorithm, encryption computes C=E_k(P), and decryption recovers P=D_k(C). The five conventional symmetric ingredients are plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. Cryptanalysis seeks recovery or other breaks without the authorized secret; cryptology includes cryptography and cryptanalysis. Steganography hides that a message exists; encryption hides its meaning.

Study-note section SEC12

> Symmetric cryptography is not inherently less secure than public-key cryptography, and speed ratios are not fixed constants.

Study-note section SEC12

> Security depends on algorithm, key size, randomness, mode and implementation.

Question sources

Computer Security review questions · Question 24

Question 145 | NS0441

In DES, what is the block size and key size?

Hint

Track block width, effective key bits and the transformations within a round.

Answer: DES uses 64-bit data blocks and 56 effective key bits. A commonly stored 64-bit key representation includes eight parity bits, which do not increase the effective key space. DES has 16 Feistel rounds and is a legacy algorithm.

SEC17 distinguishes block width from effective key entropy and parity representation. Calling DES a 64-bit-key security level would count nonsecret parity bits as independent choices. The note also warns that DES/triple DES are legacy mechanisms rather than recommendations for new protection.

Primary note proof

Study-note section SEC17

> DES has a 64-bit block, 56 effective key bits (eight other bits serve parity), and 16 Feistel rounds. After initial permutation, each round computes L_i=R_(i-1), R_i=L_(i-1) XOR f(R_(i-1),K_i). Its right-half expansion makes 48 bits, XOR mixes a 48-bit subkey, eight S-boxes each map six bits to four, and permutation rearranges the 32 output bits. The outer S-box input bits select a row and the inner four select a column.

Study-note section SEC17

> DES and triple DES are legacy algorithms, not choices for new protection.

Study-note section SEC17

> Triple DES repeats DES operations but does not provide 168-bit effective security merely because three 56-bit keys are supplied.

Question sources

Computer Security review questions · Question 25

Question 146 | NS0442

Why does DES divide the plaintext into left and right halves?

Hint

Track block width, effective key bits and the transformations within a round.

Answer: DES splits its 64-bit block into two 32-bit halves for Feistel rounds: L_i=R_(i-1) and R_i=L_(i-1) XOR f(R_(i-1),K_i). This permits inversion using the same overall structure with reversed round keys, without requiring f to be invertible.

SEC17 gives the round equations and inversion property. The old right half becomes the new left half; the old left is mixed with a keyed transformation of the old right. Keeping those relationships allows recovery by reversing the key order rather than independently inverting every S-box/transformation.

Primary note proof

Study-note section SEC17

> DES has a 64-bit block, 56 effective key bits (eight other bits serve parity), and 16 Feistel rounds. After initial permutation, each round computes L_i=R_(i-1), R_i=L_(i-1) XOR f(R_(i-1),K_i). Its right-half expansion makes 48 bits, XOR mixes a 48-bit subkey, eight S-boxes each map six bits to four, and permutation rearranges the 32 output bits. The outer S-box input bits select a row and the inner four select a column.

Study-note section SEC17

> Feistel inversion uses the same overall structure with reversed round keys; f need not itself be invertible.

Study-note section SEC17

> Triple DES repeats DES operations but does not provide 168-bit effective security merely because three 56-bit keys are supplied.

Question sources

Computer Security review questions · Question 26

Question 147 | NS0443

What is the purpose of the S-box in DES?

Hint

Track block width, effective key bits and the transformations within a round.

Answer: DES S-boxes provide nonlinear substitution: each of eight boxes maps six input bits to four output bits. The outer two bits select the row and the inner four select the column, giving 32 output bits altogether.

SEC17 identifies the S-boxes as the nonlinear component that resists simple linear descriptions. Their six-to-four mapping follows expansion and key mixing, then a permutation rearranges the outputs. They are not merely a reversible bit permutation; Feistel structure supplies overall invertibility without requiring each S-box itself to be invertible.

Primary note proof

Study-note section SEC17

> DES has a 64-bit block, 56 effective key bits (eight other bits serve parity), and 16 Feistel rounds. After initial permutation, each round computes L_i=R_(i-1), R_i=L_(i-1) XOR f(R_(i-1),K_i). Its right-half expansion makes 48 bits, XOR mixes a 48-bit subkey, eight S-boxes each map six bits to four, and permutation rearranges the 32 output bits. The outer S-box input bits select a row and the inner four select a column.

Study-note section SEC17

> The nonlinear S-boxes resist simple linear descriptions.

Study-note section SEC17

> Feistel inversion uses the same overall structure with reversed round keys; f need not itself be invertible.

Question sources

Computer Security review questions · Question 27

Question 148 | NS0444

In RSA, what are the formulas for encryption and decryption?

Hint

Check key-generation conditions as well as modular arithmetic.

Answer: Textbook RSA encrypts c=m^e mod n and decrypts m=c^d mod n, for 0≤m<n. Here n=pq, e is coprime to φ=(p−1)(q−1), and d=e^−1 mod φ. Real RSA additionally requires appropriate standardized padding.

SEC19 supplies the formulas and their key-generation preconditions. The modular-inverse condition is necessary for a valid key pair, so arbitrary e values are not acceptable. These arithmetic expressions teach RSA operations; raw unpadded modular exponentiation is not a complete secure deployment recipe.

Primary note proof

Study-note section SEC19

> For textbook RSA choose distinct primes p,q; n=pq and φ=(p-1)(q-1). Choose 1<e<φ with gcd(e,φ)=1, then compute d=e^-1 modφ. Public key is (n,e), private includes d. Encrypt 0≤m<n by c=m^e modn; decrypt by m=c^d modn. Use modular repeated squaring for large exponents. Real RSA requires appropriate standardized padding; raw arithmetic is not a secure deployment recipe.

Study-note section SEC19

> With p=3,q=13,e=3,m=5, 5^3 mod39=8 arithmetically, but gcd(3,24)=3, so e has no inverse and this is not a valid RSA key.

Study-note section SEC19

> Check key validity before treating arithmetic results as a working encryption/decryption pair.

Question sources

Computer Security review questions · Question 28

Question 149 | NS0445

In textbook RSA key generation with p=7, q=19 and φ=108, choose the smallest integer e greater than 1 satisfying 1<e<φ and gcd(e,φ)=1. What is e?

Hint

Check key-generation conditions as well as modular arithmetic.

Answer: e = 5

The exponent must be greater than one, below 108 and coprime to 108. Testing candidates in increasing order, 2 shares factor 2, 3 shares factor 3 and 4 shares factor 2 with 108. Five shares neither prime factor of 108, so gcd(5,108)=1. It is therefore the smallest admissible exponent. Other values can also be admissible, but the explicit smallest-value rule makes this open-ended answer unique.

Primary note proof

Study-note section SEC19

> For textbook RSA choose distinct primes p,q; n=pq and φ=(p-1)(q-1). Choose 1<e<φ with gcd(e,φ)=1, then compute d=e^-1 modφ. Public key is (n,e), private includes d. Encrypt 0≤m<n by c=m^e modn; decrypt by m=c^d modn. Use modular repeated squaring for large exponents. Real RSA requires appropriate standardized padding; raw arithmetic is not a secure deployment recipe.

Study-note section SEC19

> With p=7,q=19, n=133 and φ=108, e=5 is the smallest admissible e>1, but 7 and other coprime choices also work.

Study-note section SEC19

> With p=3,q=13,e=3,m=5, 5^3 mod39=8 arithmetically, but gcd(3,24)=3, so e has no inverse and this is not a valid RSA key.

Question sources

Computer Security review questions · Question 29

Question 150 | NS0446

What is the Playfair cipher and what unique element does its grid contain?

Hint

Identify the cipher's unit, key structure and assumptions before judging its strength.

Answer: Playfair is a digraph substitution cipher using a keyed 5×5 letter grid, commonly combining I and J into one cell. It encrypts pairs by same-row, same-column or rectangle rules; fillers and padding must be specified.

SEC14 explains the 25-cell grid and letter-pair operations. Combining I/J is the conventional element that fits the alphabet into the grid, not a claim about numeric cells. Repeated-letter fillers and final padding affect actual ciphertext, so computations require those conventions rather than guessing them.

Primary note proof

Study-note section SEC14

> Playfair uses a keyed 5×5 grid, commonly combining I/J, and encrypts letter pairs. Specify duplicate-letter fillers and padding. Same row moves right, same column moves down, and rectangle pairs take the other column in their original row; decryption reverses shifts.

Question sources

Computer Security review questions · Question 30

Question 151 | NS0447

What is ARP Spoofing and which OSI layer does it target?

Hint

Identify the trust claim or communication mapping being abused.

Answer: ARP spoofing supplies false local IPv4-to-MAC associations, directing traffic toward an unintended link endpoint. It targets local link-address resolution at the network/data-link boundary and is commonly treated as a data-link-layer attack.

SEC05 identifies the poisoned association rather than DNS names or transport state. Its scope is local IPv4-to-MAC resolution, so describing it simply as arbitrary IP routing would miss the mechanism. The layer answer refers to manipulating the local link mapping, while ARP connects network and link addressing.

Primary note proof

Study-note section SEC05

> Masquerade claims another identity. ARP spoofing poisons local IPv4-to-MAC associations; DNS spoofing/cache poisoning supplies false name resolutions; IP spoofing falsifies a source address. An intermediary attack intercepts and can alter exchanges. A replay reuses a valid old message; nonces, timestamps and sequence checks can reject it. Session hijacking steals or predicts a valid session token. Protect transport, regenerate session IDs after login, restrict cookies and check authorization server-side on every sensitive request.

Question sources

Computer Security review questions · Question 31

Question 152 | NS0448

What three primary services does IPSec provide?

Hint

Match the communication task to the protocol and trace key handling separately from signing.

Answer: The principal IPsec services are confidentiality, integrity and origin authentication for IP traffic. They depend on the chosen protocol/association: AH supplies integrity/authentication without encryption, while ESP can supply confidentiality and integrity; IKE negotiates associations and keys.

IPsec operates at the network layer, with distinct AH, ESP and IKE roles. Listing the service set must not imply every IPsec packet is encrypted or that IKE itself carries protected application content. Transport/tunnel mode describes coverage/encapsulation rather than replacing the service-selection and key assumptions.

Primary note proof

Study-note section SEC22

> SSH protects remote login, commands and file transfer such as SFTP; it supports host keys and several client-authentication methods. IPsec protects IP traffic at network layer. AH provides integrity/origin authentication, not encryption; ESP can provide confidentiality and integrity; IKE negotiates security associations and keys. Transport mode protects upper-layer payload; tunnel mode encapsulates an inner IP packet inside an outer header. A VPN is a logical private connection; its actual security depends on the chosen authenticated tunnel.

Question sources

Computer Security review questions · Question 32

Question 153 | NS0449

How does a TCP SYN flood attack work?

Hint

Match the resource bottleneck or wireless mechanism to the relevant protection layer.

Answer: A TCP SYN flood creates many incomplete connection handshakes, occupying server state and exhausting resources needed for legitimate connections. Rate limits, SYN cookies, filtering or upstream controls address different parts of that resource bottleneck.

SEC06 defines the attack by incomplete-handshake state consumption. The goal is denial of usable service, not necessarily gaining control of the target. Defenses must match the constrained resource and traffic path; no single listed measure guarantees resistance to every distributed load or bandwidth bottleneck.

Primary note proof

Study-note section SEC06

> DoS exhausts or disables a resource; distributed DoS uses multiple sources, often bots or reflected/amplified responses. A SYN flood leaves many incomplete TCP handshakes occupying state. DNS flooding targets query capacity; NXDOMAIN attacks generate queries for nonexistent names to stress resolvers. Fragment floods abuse reassembly or packet-processing resources. Rate limits, SYN cookies, filtering and upstream capacity controls address different bottlenecks.

Study-note section SEC03

> Passive attacks observe without changing the target exchange: sniffing and traffic analysis.

Study-note section SEC06

> Rogue access points imitate or bypass approved wireless service.

Question sources

Computer Security review questions · Question 33

Question 154 | NS0450

What is the difference between SSL and TLS?

Hint

Distinguish peer identity, channel protection and risks outside that channel.

Answer: TLS is the successor to obsolete SSL and provides an authenticated protected channel. Handshakes negotiate parameters, authenticate peers as required and establish keys; symmetric authenticated record protection carries data. Exact handshake messages depend on TLS version.

SEC21 warns against treating an old SSL/ChangeCipherSpec narrative as the universal TLS1.3 sequence. The useful distinction is protocol succession and version-specific behavior, not a claim that the names are interchangeable or that every deployment of either is automatically secure.

Primary note proof

Study-note section SEC21

> A certificate binds a public key to a named subject through an issuer signature. Validate the chain to a trusted root, intended hostname, time validity and applicable key usage/revocation policy. A genuine certificate is not proof that a business is honest.

Study-note section SEC21

> Exact messages depend on TLS version: the old ChangeCipherSpec narrative is not a universal TLS1.3 sequence.

Study-note section SEC21

> TLS succeeds obsolete SSL and supplies an authenticated protected channel.

Question sources

Computer Security review questions · Question 34

Question 155 | NS0451

What does HTTPS stand for and what security does it add over HTTP?

Hint

Distinguish peer identity, channel protection and risks outside that channel.

Answer: HTTPS is secure HTTP: HTTP over TLS. It adds authenticated channel protection for HTTP paths, headers and content. It does not conceal all metadata, such as endpoint IPs/timing, or protect plaintext after a compromised endpoint decrypts it.

SEC21 identifies the composition and coverage. Protection depends on authentication and certificate validation rather than the mere appearance of the URL scheme. It concerns the communication channel; endpoint compromise and visible traffic metadata remain outside that confidentiality boundary.

Primary note proof

Study-note section SEC21

> A certificate binds a public key to a named subject through an issuer signature. Validate the chain to a trusted root, intended hostname, time validity and applicable key usage/revocation policy. A genuine certificate is not proof that a business is honest.

Study-note section SEC21

> It protects HTTP paths, headers and content on the channel, but does not hide all metadata such as IP endpoints and traffic timing, nor protect data after a compromised endpoint decrypts it.

Study-note section SEC21

> HTTPS is HTTP over TLS.

Question sources

Computer Security review questions · Question 35

Question 156 | NS0452

What is DNS Spoofing?

Hint

Identify the trust claim or communication mapping being abused.

Answer: DNS spoofing provides false name-resolution data, potentially redirecting a requested domain to an attacker-chosen destination. Cache poisoning is a form that places false answers in a resolver cache. DNSSEC authenticates DNS data when appropriately deployed and validated.

SEC05 distinguishes false DNS mapping from ARP/IP spoofing. SEC06 separates DNS flooding, which consumes query capacity, from answer authenticity. A successful-looking name lookup can still be maliciously wrong; checking that a response exists is different from establishing its trusted origin and integrity.

Primary note proof

Study-note section SEC05

> Masquerade claims another identity. ARP spoofing poisons local IPv4-to-MAC associations; DNS spoofing/cache poisoning supplies false name resolutions; IP spoofing falsifies a source address. An intermediary attack intercepts and can alter exchanges. A replay reuses a valid old message; nonces, timestamps and sequence checks can reject it. Session hijacking steals or predicts a valid session token. Protect transport, regenerate session IDs after login, restrict cookies and check authorization server-side on every sensitive request.

Study-note section SEC06

> DNS flooding targets query capacity; NXDOMAIN attacks generate queries for nonexistent names to stress resolvers.

Study-note section SEC06

> Match a defense to its layer: authenticated IPsec can resist IP impersonation, DNSSEC authenticates DNS data, TLS authenticates/protects application transport, and bounds-safe code prevents overflows.

Question sources

Computer Security review questions · Question 36

Question 157 | NS0453

What is the purpose of SSH (Secure Shell)?

Hint

Match the communication task to the protocol and trace key handling separately from signing.

Answer: SSH (Secure Shell) protects remote login, command execution and file transfer such as SFTP. It uses host keys and supports several client-authentication methods. Correct host-key validation and restricted management access remain important.

SEC22 states the protected remote-administration functions. SSH is not merely encryption of a password; it protects the session and requires authentic peer identity. File transfer through SFTP is related to SSH rather than being ordinary FTP with a new name.

Primary note proof

Study-note section SEC22

> SSH protects remote login, commands and file transfer such as SFTP; it supports host keys and several client-authentication methods. IPsec protects IP traffic at network layer. AH provides integrity/origin authentication, not encryption; ESP can provide confidentiality and integrity; IKE negotiates security associations and keys. Transport mode protects upper-layer payload; tunnel mode encapsulates an inner IP packet inside an outer header. A VPN is a logical private connection; its actual security depends on the chosen authenticated tunnel.

Study-note section SEC-S03

> SSH expands to Secure Shell and provides secure remote login and related network services.

Study-note section SEC-S03

> AAA names authentication, authorization and accounting functions, rather than a particular remote-shell channel.

Question sources

Computer Security review questions · Question 37

Question 158 | NS0454

What is a Man-in-the-Middle (MITM) attack?

Hint

Identify the trust claim or communication mapping being abused.

Answer: A man-in-the-middle attack places an attacker between communicating parties to intercept and potentially alter their exchange. Authentication of the intended peer and a protected channel are needed; encryption without authentic key binding can still allow intermediary substitution.

SEC05 gives the interception/alteration mechanism, while SEC18's unauthenticated Diffie-Hellman example explains why secrecy mechanisms alone are insufficient. The attacker may relay messages so both victims believe they communicate directly, making peer authentication and correct certificate/key checks essential.

Primary note proof

Study-note section SEC05

> Masquerade claims another identity. ARP spoofing poisons local IPv4-to-MAC associations; DNS spoofing/cache poisoning supplies false name resolutions; IP spoofing falsifies a source address. An intermediary attack intercepts and can alter exchanges. A replay reuses a valid old message; nonces, timestamps and sequence checks can reject it. Session hijacking steals or predicts a valid session token. Protect transport, regenerate session IDs after login, restrict cookies and check authorization server-side on every sensitive request.

Study-note section SEC18

> Diffie-Hellman agrees on a secret but unauthenticated DH permits intermediary substitution.

Study-note section SEC18

> Hybrid systems establish or encapsulate a symmetric secret with public-key mechanisms, then protect bulk data symmetrically.

Question sources

Computer Security review questions · Question 38

Question 159 | NS0455

What is the SET (Secure Electronic Transaction) protocol and who developed it?

Hint

Match the communication task to the protocol and trace key handling separately from signing.

Answer: SET means Secure Electronic Transaction. It was a Visa/Mastercard-led specification for protecting payment-card transactions with certificates and signatures. It is distinct from an ordinary HTTPS connection or a general payment-processing system.

SEC22 supplies the expansion, origin and protection role. Certificates/signatures support authenticated transaction relationships rather than defining the entire financial settlement infrastructure. Calling every TLS-protected checkout SET would conflate separate protocols with different scope and structure.

Primary note proof

Study-note section SEC22

> PGP/OpenPGP combines symmetric content protection with public-key key handling and signatures. S/MIME uses certificates for signed and/or enveloped MIME content. Signatures can be sent separately/clear-signed without confidentiality; enveloped data encrypts content and wraps content keys for recipients. SET, Secure Electronic Transaction, was a Visa/Mastercard-led specification for protecting card transactions using certificates and signatures; it is distinct from a payment-processing system or ordinary HTTPS.

Study-note section SEC21

> TLS succeeds obsolete SSL and supplies an authenticated protected channel.

Study-note section SEC21

> HTTPS is HTTP over TLS.

Question sources

Computer Security review questions · Question 39

Question 160 | NS0456

What is Session Hijacking?

Hint

Identify the trust claim or communication mapping being abused.

Answer: Session hijacking is taking over an existing authenticated session by stealing or predicting a valid session token. Defenses include protected transport, regenerating IDs after login, restrictive cookie handling and server-side authorization on sensitive requests.

SEC05 identifies token possession/prediction as the takeover path. Establishing a secure initial login does not alone protect the later bearer token, and checking authorization on every sensitive action remains necessary. The mechanism differs from replaying an old message, although stolen session material can enable reused requests.

Primary note proof

Study-note section SEC05

> Masquerade claims another identity. ARP spoofing poisons local IPv4-to-MAC associations; DNS spoofing/cache poisoning supplies false name resolutions; IP spoofing falsifies a source address. An intermediary attack intercepts and can alter exchanges. A replay reuses a valid old message; nonces, timestamps and sequence checks can reject it. Session hijacking steals or predicts a valid session token. Protect transport, regenerate session IDs after login, restrict cookies and check authorization server-side on every sensitive request.

Question sources

Computer Security review questions · Question 40

Question 161 | NS0457

What is a firewall and what are its three characteristics as a 'reference monitor'?

Hint

Separate security-policy enforcement and assurance from ordinary OS features.

Answer: A firewall enforces traffic policy on flows it mediates. As a reference-monitor design, enforcement must cover every relevant access, resist tampering and be small/simple enough to analyze. For a firewall this is its traffic boundary, not every host-memory operation.

SEC28 defines the three reference-monitor requirements and explicitly limits a firewall to traversing traffic. SEC23 describes policy enforcement and bypass limits. Thus complete mediation requires preventing alternate paths within the claimed boundary; deploying a firewall does not make it a universal host security kernel.

Primary note proof

Study-note section SEC28

> A reference monitor is an abstract access-checking mechanism: it must mediate every relevant access, resist tampering and be sufficiently small/simple to analyze. A security kernel implements core enforcement. A firewall can mediate its traversing traffic but cannot enforce every host-memory access. Assurance is justified confidence that controls implement the claims; functionality is what controls claim to do. A GUI or multitasking support alone does not establish trust.

Study-note section SEC23

> A firewall enforces traffic policy.

Study-note section SEC23

> Bypass paths, insiders, encrypted uninspected content and infected removable media limit firewall protection.

Question sources

Computer Security review questions · Question 41

Question 162 | NS0458

What is the key difference between a Packet-Filtering firewall and a Stateful Inspection firewall?

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer: A stateless packet-filtering firewall matches each packet’s header fields such as addresses, protocol and ports. Stateful inspection also tracks permitted connection context, allowing decisions based on whether traffic belongs to an allowed exchange.

SEC23 distinguishes header matching from connection-aware inspection. Both can filter ports, so the difference is not port-filtering capability alone. State tracking does not imply unlimited application-content understanding or automatic encryption; those are additional, separate functions.

Primary note proof

Study-note section SEC23

> A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Question sources

Computer Security review questions · Question 42

Question 163 | NS0459

What is an Application-Level Gateway (proxy firewall)?

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer: An application-level gateway/proxy terminates and relays application exchanges on behalf of clients/services and can inspect application content. It differs from a simple header packet filter or circuit relay that lacks full application-content interpretation.

SEC23 defines the proxy by its application mediation boundary. It can enforce application-specific policy because it participates in the exchanges rather than merely routing packets. That role does not inherently mean encryption or protection against traffic that bypasses the proxy.

Primary note proof

Study-note section SEC23

> A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Question sources

Computer Security review questions · Question 43

Question 164 | NS0460

What is the difference between an IDS and an IPS?

Hint

Separate observation from blocking and sensor placement from detection method.

Answer: An IDS detects suspicious activity and alerts. An IPS can enforce prevention/blocking, usually on the traffic path. Both need contextual analysis because false positives and missed attacks remain possible.

SEC24 makes action the distinction, independently of host/network sensor location or signature/anomaly detection method. Observing a suspected attack does not itself prevent it. Inline blocking can affect legitimate service when an alert is false, so evidence and policy determine the appropriate response.

Primary note proof

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline. Host-based sensors examine host events, network-based sensors inspect traffic. Signature detection recognizes known patterns; anomaly detection compares with a baseline and may detect novel behavior. Neither guarantees every attack is detected; anomaly detection can raise many false positives, and signatures can sometimes match variants.

Study-note section SEC24

> A true positive is a real attack detected, false negative is a missed attack, false positive is innocent activity flagged, and true negative is innocent activity accepted.

Study-note section SEC24

> Obfuscation makes code harder to understand and can hamper analysis, but does not reliably protect embedded secrets or remove vulnerabilities.

Question sources

Computer Security review questions · Question 44

Question 165 | NS0461

What are the two main types of IDS detection methods?

Hint

Separate observation from blocking and sensor placement from detection method.

Answer: The two main detection methods are signature-based and anomaly-based. Signatures recognize known patterns; anomaly methods compare behavior with a baseline and may find novel deviations. Both can miss attacks or produce false positives.

SEC24 explicitly gives these methods and their limitations. A deviation is not automatically malicious, and a matching signature does not prove every attack variant is captured. Host-based versus network-based describes sensor placement, a different classification from how the detection decision is made.

Primary note proof

Study-note section SEC24

> An IDS observes and alerts; an IPS can enforce blocking, usually inline. Host-based sensors examine host events, network-based sensors inspect traffic. Signature detection recognizes known patterns; anomaly detection compares with a baseline and may detect novel behavior. Neither guarantees every attack is detected; anomaly detection can raise many false positives, and signatures can sometimes match variants.

Study-note section SEC24

> Detecting 90 of 100 spam messages with five innocent messages flagged gives recall90%, precision90/95≈94.7%.

Study-note section SEC24

> A true positive is a real attack detected, false negative is a missed attack, false positive is innocent activity flagged, and true negative is innocent activity accepted.

Question sources

Computer Security review questions · Question 45

Question 166 | NS0462

What limitation does a firewall have regarding data outside its perimeter?

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer: A firewall cannot enforce policy on data or attack paths that bypass it. Insiders, removable media and uninspected encrypted content can evade its protection, so outside/perimeter-bypassing data needs endpoint, access and other controls.

SEC23 confines firewall protection to traffic actually mediated and inspectable under its design. A boundary rule does not sanitize every file after it arrives through another path. This is a scope limitation rather than proof that firewalls are useless or that all encrypted traffic is malicious.

Primary note proof

Study-note section SEC23

> Host firewalls protect individual machines; network firewalls segment networks and can create a DMZ for public-facing services. Rules should be explicit, ordered as implemented, least-privilege and logged appropriately. Stateful and next-generation devices can also filter ports; port filtering is not exclusive to one firewall category. Bypass paths, insiders, encrypted uninspected content and infected removable media limit firewall protection. Hardware firewall does not imply any fixed number of ports.

Question sources

Computer Security review questions · Question 46

Question 167 | NS0463

What is PGP (Pretty Good Privacy) and what security services does it provide?

Hint

Match the communication task to the protocol and trace key handling separately from signing.

Answer: PGP (Pretty Good Privacy)/OpenPGP combines symmetric encryption for content, public-key handling of content keys and digital signatures. It can provide confidentiality and integrity/origin-authentication evidence according to which operations are used and how keys are validated.

SEC22 gives the hybrid composition and SEC20 explains signature scope. A signature alone does not conceal content; encryption alone is not the same as a validated signature. Recipient/signer key authenticity and custody therefore matter to the protection actually obtained.

Primary note proof

Study-note section SEC22

> PGP/OpenPGP combines symmetric content protection with public-key key handling and signatures. S/MIME uses certificates for signed and/or enveloped MIME content. Signatures can be sent separately/clear-signed without confidentiality; enveloped data encrypts content and wraps content keys for recipients. SET, Secure Electronic Transaction, was a Visa/Mastercard-led specification for protecting card transactions using certificates and signatures; it is distinct from a payment-processing system or ordinary HTTPS.

Study-note section SEC-S03

> PGP expands to Pretty Good Privacy; its message-protection design combines conventional and public-key encryption with digital signatures.

Study-note section SEC20

> Verification checks origin/integrity under key-identity and key-custody assumptions; it does not conceal content.

Question sources

Computer Security review questions · Question 47

Question 168 | NS0464

What is S/MIME and how does 'Signed and Enveloped Data' work?

Hint

Match the communication task to the protocol and trace key handling separately from signing.

Answer: S/MIME protects MIME messages using certificates for signatures and/or enveloped encryption. In sign-then-encrypt, sign the MIME entity, package its content/signature, encrypt it with a fresh symmetric content key and protect that key for each recipient. The recipient recovers the key, decrypts, then verifies with the validated signer certificate.

SEC22 specifies this construction while allowing other nesting orders with different coverage. Signed data supplies origin/integrity evidence without inherently concealing content; enveloped data encrypts content and wraps keys. Signing is not generic private-key encryption of the whole message, and certificate/key validation is essential to interpreting verification.

Primary note proof

Study-note section SEC22

> PGP/OpenPGP combines symmetric content protection with public-key key handling and signatures. S/MIME uses certificates for signed and/or enveloped MIME content. Signatures can be sent separately/clear-signed without confidentiality; enveloped data encrypts content and wraps content keys for recipients. SET, Secure Electronic Transaction, was a Visa/Mastercard-led specification for protecting card transactions using certificates and signatures; it is distinct from a payment-processing system or ordinary HTTPS.

Study-note section SEC22

> A recipient recovers the key, decrypts the signed entity and verifies its signature using the validated signer certificate/public key.

Study-note section SEC22

> One signed-and-encrypted S/MIME construction first signs the MIME entity, packages its content and signature, then encrypts that signed entity with a fresh symmetric content key.

Related reading (context, not primary note proof)

Study-note section SEC20

Study-note section SEC21

Question sources

Computer Security review questions · Question 48

Question 169 | NS0465

What is a Circuit-Level Gateway and what OSI layer does it operate at?

Hint

Identify which traffic is mediated and what information drives the device's decisions.

Answer: A circuit-level gateway relays connections, commonly using SOCKS, without full application-content interpretation. Textbook OSI mappings associate it with session/transport functions rather than a universally fixed physical OSI device layer. It need not encrypt traffic or constitute a VPN.

A circuit gateway relays connections, while application proxies inspect content and packet filters match headers. The session/transport layer label is a teaching mapping; the relay function is the stable distinction. Unspecified circuit gateways must not be assumed to supply authenticated encrypted tunnels.

Primary note proof

Study-note section SEC23

> A firewall enforces traffic policy. Stateless packet filtering matches headers such as source/destination, protocol and ports. Stateful inspection tracks permitted connection context; application proxies terminate and relay application exchanges and can inspect their content. A circuit gateway relays connections, commonly SOCKS, without full application-content interpretation; textbooks associate it with session/transport functions. Such a gateway need not encrypt and is not automatically a VPN.

Question sources

Computer Security review questions · Question 49

Question 170 | NS0466

In the RSA example where p=3, q=13, and public key e=3, what is the ciphertext for plaintext m=5?

Hint

Check key-generation conditions as well as modular arithmetic.

Answer: The modular exponentiation gives c=5^3 mod(3×13)=125 mod39=8. However φ=(3−1)(13−1)=24 and gcd(3,24)=3, so e=3 has no modular inverse. This is an arithmetic result, not a valid RSA encryption/decryption key pair.

SEC19 explicitly identifies the defective source example. Showing ciphertext 8 must not endorse the key setup: RSA requires e coprime to φ so a private exponent can invert it. The calculation and the validity check answer different parts of the example, and both must be preserved.

Primary note proof

Study-note section SEC19

> For textbook RSA choose distinct primes p,q; n=pq and φ=(p-1)(q-1). Choose 1<e<φ with gcd(e,φ)=1, then compute d=e^-1 modφ. Public key is (n,e), private includes d. Encrypt 0≤m<n by c=m^e modn; decrypt by m=c^d modn. Use modular repeated squaring for large exponents. Real RSA requires appropriate standardized padding; raw arithmetic is not a secure deployment recipe.

Study-note section SEC19

> With p=3,q=13,e=3,m=5, 5^3 mod39=8 arithmetically, but gcd(3,24)=3, so e has no inverse and this is not a valid RSA key.

Study-note section SEC19

> Check key validity before treating arithmetic results as a working encryption/decryption pair.

Question sources

Computer Security review questions · Question 50

Question 171 | NS0467

What is the Hill Cipher and why is it considered strong?

Hint

Identify the cipher's unit, key structure and assumptions before judging its strength.

Answer: Hill is a polygraphic matrix substitution cipher, C=KP mod26. Its key must be invertible modulo 26, with determinant coprime to 26. It is not strong by modern standards: enough independent known plaintext/ciphertext blocks reveal the key through its linear structure.

SEC14 rejects the supplied claim of known-plaintext resistance. Combining letters into vectors does not remove linear algebraic relationships, and a large-looking matrix key is not proof of modern security. The correct answer explains the construction while correcting the question's implied blanket strength claim.

Primary note proof

Study-note section SEC14

> Hill uses C=KP mod26 for letter vectors. K needs an inverse modulo26, equivalently determinant coprime to26. It is linear: enough independent known plaintext/ciphertext blocks reveal K=CP^-1 mod26. Hill is vulnerable to known-plaintext attacks; polygraphic substitution alone is not modern strength.

Question sources

Computer Security review questions · Question 51

Network and System Administration

Questions & Worked Answers

101 questions

Question 1 | Q00011

You administer a network and need to display active connections with their protocol, local address and port, foreign address and port, and connection state. Which command provides this connection table?

A. traceroute

B. ipconfig

C. netstat

D. nslookup

Hint

Match the requested observation to the tool’s actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer C: netstat.

NSA07 identifies netstat as the tool that displays connections, protocol names, local and foreign endpoints, and TCP connection states. That is the complete set of information requested, so option C is right. Numeric-output flags can keep addresses and ports numeric; the Windows form netstat -ano also includes process IDs.

Why each choice is right or wrong

A - Incorrect: Traceroute reports responding hops on a network path, not a connection table with local/foreign ports and TCP states.

B - Incorrect: Ipconfig reports adapter addressing and configuration, not active transport connections and their remote endpoints.

C - Correct: Netstat displays the requested protocol, endpoint and connection-state information.

D - Incorrect: Nslookup queries DNS records rather than listing active connections or TCP states.

Primary note proof

Study-note section NSA07

> netstat displays connections/endpoints/protocols and TCP states; netstat -ano displays numeric addresses/ports and PIDs. Use the exact command name netstat rather than netstart. ping probes IP reachability; traceroute/tracert reveals responding hops using TTL/hop-limit expiry. ICMP filtering can prevent replies without proving the service is down. nslookup inspects DNS.

Study-note section NSA07

Windows ipconfig shows addresses, mask and gateway; ipconfig /all adds complete adapter configuration including physical/MAC address, DHCP and DNS information. /renew requests renewal, /release releases a dynamic address and /flushdns clears the client DNS cache. Linux ip addr and ip route show interface addresses and routes; exact persistent configuration varies by distribution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 2-7

TCP and UDP must use port numbers to communicate with the upper layers, because they‘re what keep track of different conversations crossing the network simultaneously. These port numbers identify the source and destination application or process in the TCP segment. There are 216 = 65,536 ports available.

Data Communication and Computer Networking module | PDF page 27 | lines 29-31

Traceroute Using ICMP timeouts, Traceroute is used to discover the path a packet takes as it traverses an Internetwork.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 11

Question 2 | Q00077

Which statements about ordinary dynamic DHCP allocation are incorrect? Select all that apply.

A. An IP leased by the DHCP for a host is permanent

B. DHCP assures connectivity ever users move from location to location

C. The IP address are chosen from a predetermined range of address

D. DHCP allows a host to acquire an IP address automatically

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A, B: An IP leased by the DHCP for a host is permanent; DHCP assures connectivity ever users move from location to location

A dynamic DHCP lease lasts for a specified interval and must be renewed; it is not a permanent assignment. DHCP can automate address configuration after moving, but connectivity also depends on a reachable server or relay, a suitable scope and a working network. The server selects addresses from its configured pool. Thus the permanence and guaranteed-connectivity claims are false, while pool selection and automatic acquisition are true.

Why each choice is right or wrong

A - Correct: A leased dynamic address is time-limited and can become reusable after expiration; it is not permanent.

B - Correct: DHCP alone cannot assure connectivity after every location change; server, relay and network conditions matter.

C - Incorrect: A DHCP server manages an address scope/pool, supporting selection from a predetermined range.

D - Incorrect: Automatic acquisition of an address and options is a supported DHCP function.

Primary note proof

Study-note section NSA04

> DHCP supplies addresses and configuration options. Typical initial IPv4 allocation is Discover→Offer→Request→ACK. The server manages its scope/pool, allocated leases, exclusions and reservations; options may include mask, gateway and DNS servers. A dynamic lease lasts for a specified interval, not forever.

Study-note section NSA05

> Use a scope matching the subnet, exclude manually assigned server/router addresses, set correct options and provide a relay when client broadcasts must cross a router. Static assignments and DHCP can coexist; avoid address conflicts. DHCP lowers repetitive administration but availability/security depend on deployment and protection from rogue replies. It does not guarantee connectivity whenever a device changes location.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

> The Process/Application Layer Protocols
> • Telnet
> • File Transfer Protocol (FTP)
> • Network File System (NFS).
> • Simple Mail Transfer Protocol (SMTP
> • POP3 is used to receive mail.
> • Simple Network Management Protocol (SNMP)
> • Domain Name Service (DNS)
> • Dynamic Host Configuration Protocol (DHCP)

Question sources

MOE Computer Science Exit Exam · 2016 · Question 77

Question 3 | Q00091

Which listed operation is not a DHCP server function?

A. It grants an IP address when receives a request from a client.

B. It maintains the information about client configuration parameters.

C. It maintains a database of available IP addresses.

D. Resolve hostnames to IP addresses using DNS records.

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: Resolve hostnames to IP addresses using DNS records.

A DHCP server manages address pools, lease allocations and configuration parameters supplied to clients. It may allocate an address in response to a client exchange and retain information about availability and options. Hostname-to-address resolution is a DNS function instead. Repairing the duplicated malformed option supplies a distinct nonfunction while preserving the original distinction between automatic configuration and other network services.

Why each choice is right or wrong

A - Incorrect: Allocating an address in response to client requests is a DHCP server function.

B - Incorrect: The server manages lease and configuration parameters, so this is not a nonfunction.

C - Incorrect: Maintaining scope/pool availability is explicitly part of DHCP server management.

D - Correct: Hostname-to-address resolution belongs to DNS rather than DHCP allocation.

Primary note proof

Study-note section NSA04

> DHCP supplies addresses and configuration options. Typical initial IPv4 allocation is Discover→Offer→Request→ACK. The server manages its scope/pool, allocated leases, exclusions and reservations; options may include mask, gateway and DNS servers. A dynamic lease lasts for a specified interval, not forever.

Study-note section NSA06

> Forward resolution obtains an address from a name, using A for IPv4 and AAAA for IPv6. Reverse lookup uses a PTR record in the reverse namespace to obtain a name from an address. CNAME aliases a name, MX identifies mail exchangers and NS identifies authoritative name servers. Forward and reverse records need not both exist or be perfect inverses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

MOE Computer Science Exit Exam · 2016 · Question 91

Question 4 | Q00185

Based on the predefined policy of Network management, controlling access to the network is the task of

A. Fault Management

B. Performance Management

C. Security Management

D. Configuration Management

E. None

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer C: Security Management

PDF p.30 places restricting access to authorized users under security procedures. The requested access-control responsibility therefore matches Security Management.

Why each choice is right or wrong

A - Incorrect: Fault management concerns failures rather than the access authorization described on p.30.

B - Incorrect: Capacity/performance is a different responsibility (p.29), not authorization.

C - Correct: Security procedures explicitly ensure only authorized users have network access (p.30).

D - Incorrect: Configuration management is not the named access-security responsibility.

E - Incorrect: The security alternative directly matches the stated responsibility.

Primary note proof

Study-note section NSA01

Use documentation, standardization, least privilege, automation and measured change. Record the current state, intended change, rollback and validation. Schedule maintenance according to service impact and urgency, not merely because a calendar date arrived. Fault management restores failures; performance management measures service behavior; configuration management tracks settings; security management controls access; accounting tracks usage.

Study-note section NSA26

A security policy states required behavior, scope, accountable roles and enforcement. Begin with assets/service objectives, assess threats and weaknesses, choose controls, define exceptions/incident procedures, approve/communicate, monitor and periodically review. CIA means confidentiality, integrity and availability; neither sensitive-data protection nor availability alone is the universal primary goal.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Question sources

Addis Ababa University mock exam · Question 87

Question 5 | Q00186

A Unix utility which used for checking and repairing file system inconsistencies.

A. init

B. halt

C. inittab

D. fsck

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: fsck

NSA20 directly defines fsck as the filesystem-checking and optional repair utility. Repairs must follow the filesystem's supported safe or offline procedure. The alternatives refer to initialization/shutdown or an initialization configuration file, not the checking-and-repair role stated in the question.

Why each choice is right or wrong

A - Incorrect: Init concerns system initialization/service management rather than the filesystem-checking utility defined in NSA20.

B - Incorrect: Halt stops a system; it is not the utility for checking and repairing filesystem inconsistencies.

C - Incorrect: Inittab is an initialization configuration file, not the filesystem-checking command.

D - Correct: Fsck checks filesystems and can invoke filesystem-specific repair tools under supported safe conditions.

Primary note proof

Study-note section NSA20

> ext2 is a Linux filesystem; ext4 is a later filesystem; HPFS is historically associated with OS/2, while swap is backing space rather than an ordinary user-file filesystem. fsck checks and optionally repairs filesystems through filesystem-specific tools; it is not init, halt or an inittab file. Repairs require the filesystem's supported offline/safe procedure. Mount attaches a filesystem to a directory; partitioning, mounting, quotas and capacity belong to storage administration.

Study-note section NSA-S03

> In a traditional SysV-style system, init starts and supervises configured processes; /etc/inittab is its configuration file. The halt command stops the system through the supported shutdown procedure. Neither initialization, shutdown nor their configuration file is a filesystem-consistency check; fsck supplies that separate checking and optional repair role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 61 | lines 26-36

> What is Unix/Linux?
> The Unix OS is a set of programs that act as a link between the computer and the user. The computer program that allocates the system resources and coordinates all the details of the computer‘s internals is called the operating system or the kernel. Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

Data Communication and Computer Networking module | PDF page 64 | lines 8-12

> /lost+found: Every partition has a lost+found for files
> that were saved during failures are here.
> /misc: For miscellaneous purposes.
> /mnt: Standard mount point for external file systems,
> e.g. a CD-ROM or a digital camera.

Question sources

Addis Ababa University mock exam · Question 88

Question 6 | Q00188

A PC obtains its IP address from a DHCP server. If the PC is taken off the network for repair, what happens to the IP address configuration?

A. The address is returned to the pool for reuse when the lease expires

B. The address lease is automatically renewed until the PC is returned

C. The configuration is held by the server to be reissued when the PC is returned

D. The configuration is permanent and nothing changes

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: The address becomes reusable when its lease expires

Taking the PC offline does not make the dynamic assignment permanent or cause endless automatic renewal. NSA04 says a disconnected client's lease can expire and the address then becomes eligible for reuse. On return, the client must validate, renew or obtain new configuration rather than assume it still owns its former address.

Why each choice is right or wrong

A - Correct: After expiry the address becomes eligible for return to the available pool, subject to server policy.

B - Incorrect: A disconnected client does not automatically renew indefinitely while it remains off the network.

C - Incorrect: The server need not hold the old address for that client after lease expiry without a reservation.

D - Incorrect: A dynamic lease has a finite interval and is not an unchanging permanent assignment.

Primary note proof

Study-note section NSA04

A client renews before expiry; it later tries rebinding if renewal fails. A disconnected host does not automatically renew forever. On lease expiry the address becomes eligible for reuse, subject to server policy. Reconnection normally triggers validation/renewal/new allocation, so an expired lease alone does not prove persistent failure. A reservation gives a predictable address to a known client while retaining DHCP configuration.

Study-note section NSA04

DHCP supplies addresses and configuration options. Typical initial IPv4 allocation is Discover→Offer→Request→ACK. The server manages its scope/pool, allocated leases, exclusions and reservations; options may include mask, gateway and DNS servers. A dynamic lease lasts for a specified interval, not forever.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Addis Ababa University mock exam · Question 90

Computer Science Exit Exam · Sene 2016 · Question 4-2

Computer Science Exit Exam practice collection · Question 161

Question 7 | Q00189

Which command in Microsoft Windows allows you to verify your IP address, subnet mask, default gateway, and MAC address?

A. C:\>ipconfig/all

B. C:\>ping

C. C:\>ipconfig

D. C:\>ipstatus

Hint

Match the requested observation to the tool's actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer A: ipconfig /all

The distinguishing requirement is the MAC address as well as IP, mask and gateway. NSA07 assigns this complete adapter detail to ipconfig /all; basic ipconfig covers the ordinary addressing summary. A displays the command with its /all switch. Ping tests reachability and ipstatus is not the defined Windows diagnostic.

Why each choice is right or wrong

A - Correct: The /all switch supplies the detailed adapter information, including the physical/MAC address requested.

B - Incorrect: Ping probes reachability rather than reporting the adapter's complete IP and MAC configuration.

C - Incorrect: Basic ipconfig reports IP, mask and gateway but omits the requested detailed physical/MAC information.

D - Incorrect: Ipstatus is not the Windows adapter-configuration command established by the complete note.

Primary note proof

Study-note section NSA07

> Windows ipconfig shows addresses, mask and gateway; ipconfig /all adds complete adapter configuration including physical/MAC address, DHCP and DNS information. /renew requests renewal, /release releases a dynamic address and /flushdns clears the client DNS cache. Linux ip addr and ip route show interface addresses and routes; exact persistent configuration varies by distribution.

Study-note section NSA07

> netstat displays connections/endpoints/protocols and TCP states; netstat -ano displays numeric addresses/ports and PIDs. Use the exact command name netstat rather than netstart. ping probes IP reachability; traceroute/tracert reveals responding hops using TTL/hop-limit expiry. ICMP filtering can prevent replies without proving the service is down. nslookup inspects DNS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 65 | lines 39-47

> Network configuration commands
> The ip Command
> The distribution-specific scripts and graphical tools
> are front-ends to ip (or ifconfig and route on older
> systems) to display and configure the kernel‘s
> networking configuration. The ip command is used for
> assigning IP addresses to interfaces, for setting up routes
> to the Internet and to other networks, for displaying
> TCP/IP

Data Communication and Computer Networking module | PDF page 27 | lines 26-28

Ping (Packet Internet Groper) uses ICMP echo messages to check the physical and logical connectivity of machineson a network.

Question sources

Addis Ababa University mock exam · Question 91

Computer Science Exit Exam · Sene 2016 · Question 29-2

Computer Science Exit Exam practice collection · Question 160

Question 8 | Q00191

You are the administrator of an active directory domain. A user complains to you that he is unable to change his password. No other users have this issue. What is the most likely cause of the problem?

A. Insufficient login credentials

B. He is a member of the Administrators group

C. He is not a member of the domain

D. The property "User cannot change password" has been enabled when it was created

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: The account’s User cannot change password property

A failure isolated to one domain user favors inspecting that user’s permissions and cannot-change-password setting. D is the specific account control matching the symptom, and NSA17 explicitly identifies it as a plausible exam cause. It remains a most-likely option among these choices, not proof of its actual state: password age/rules and other account conditions would require real diagnosis.

Why each choice is right or wrong

A - Incorrect: Generic insufficient credentials gives no specific cause for this domain user’s isolated password-change restriction.

B - Incorrect: Administrators membership does not inherently prohibit users from changing their own passwords.

C - Incorrect: The scenario concerns a domain user; lacking domain membership is not the specific account restriction supported here.

D - Correct: The cannot-change-password setting directly blocks the operation and matches the isolated account symptom.

Primary note proof

Study-note section NSA17

If only one domain user cannot change a password, inspect that account's cannot-change-password setting and relevant policy/permission before blaming all directory infrastructure. Other possibilities include minimum password age, password rules or account state; the setting is a plausible exam cause, not a conclusion from failure alone. Membership in Administrators does not inherently prohibit changing one's password. Privileged reset and user-initiated password change are different operations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 42 | lines 36-49

All user accounts have specific capabilities, privileges, and rights. When you create a user account, you can grant the user specific capabilities by making the user a member of one or more groups. This gives the user the capabilities of these groups. You then assign additional capabilities by making a user a member of the appropriate groups or withdraw capabilities by removing a user from a group. An important part of an administrator‘s job is being able to determine and set permissions, privileges, and logon rights as necessary. Although you can‘t change a group‘s built-in capabilities, you can change a group‘s default privileges and logon rights. For example, you could revoke network access to a computer by removing agroup‘s right to access the computer from the network.

Data Communication and Computer Networking module | PDF page 34 | lines 7-14

Domain Controllers
A domain controller (DC) is a server computer that responds to security authentication requests within a computer network domain. It is a network server that is responsible for allowing end devices to access shared domain resources. It authenticates users, stores user account information and enforces security policy for a domain. It is most commonly implemented in Microsoft

Question sources

Addis Ababa University mock exam · Question 93

Question 9 | Q00353

Which is the most specific DNS term for obtaining an IP address from a hostname?

A. Forward resolution

B. Backward resolution

C. Binding

D. Resolving

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: Forward resolution

Forward resolution names the specific direction from a hostname to an address record. Resolving is a broader term that also includes this operation, so it is less specific than the requested answer. Reverse lookup goes from an address to a name, and binding is not the DNS operation defined here. Asking for the most specific term makes the overlapping general description a meaningful distractor.

Why each choice is right or wrong

A - Correct: Forward resolution is the precise name-to-address operation defined by NSA06.

B - Incorrect: Reverse lookup runs in the opposite direction, obtaining a name from an address.

C - Incorrect: Binding is not the note's term for the DNS name-to-address operation; LDAP bind authenticates a session.

D - Incorrect: Resolving includes forward lookup but is broader than the most specific term requested.

Primary note proof

Study-note section NSA06

> Forward resolution obtains an address from a name, using A for IPv4 and AAAA for IPv6. Reverse lookup uses a PTR record in the reverse namespace to obtain a name from an address. CNAME aliases a name, MX identifies mail exchangers and NS identifies authoritative name servers. Forward and reverse records need not both exist or be perfect inverses.

Study-note section NSA06

> Resolvers cache answers for their TTL. Query nslookup name or specify the DNS server explicitly; compare the configured resolver with the authoritative answer when appropriate. A successful ping by address but failed name lookup suggests naming/configuration trouble. Ping by name may display a resolved address, but nslookup is the direct DNS diagnostic. "Resolving" is a general term that includes forward resolution.

Study-note section NSA14

LDAP accesses directory entries/attributes identified by distinguished names. Bind authenticates a session; search retrieves matching entries; compare tests an attribute value; add/delete/modify change entries; modify-DN renames or moves; unbind ends the session. StartTLS upgrades a connection to TLS where supported; LDAPS uses TLS from connection setup. Protect credentials and verify server identity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

/etc/hosts
The /etc/hosts file always contains the localhost IP
address, 127.0.0.1, which is
usedfor interprocess communication.
Never remove this line! Sometimes contains
addresses of additional hosts, whichcan be contacted
without using an external naming service such as
DNS (the Domain Name Server).

Question sources

Computer Science model exam · Question 84

University of Gondar model exam · Question 83

Question 10 | Q00354

You know a server hostname and want to query its DNS address records directly, optionally choosing the DNS server. Which listed diagnostic tool is designed for that task?

A. Nslookup

B. Ping command

C. Traceroute

D. Netstat

Hint

Match the requested observation to the tool’s actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer A: Nslookup

Nslookup directly queries DNS for the address records associated with a name and can use a specified DNS server. Ping may display a resolved address, but its principal task is testing IP reachability. Traceroute examines responding path hops, while netstat exposes endpoints and connection state. The direct-DNS-query requirement identifies the intended diagnostic without relying on incidental address displays from other commands.

Why each choice is right or wrong

A - Correct: Nslookup directly queries DNS records for the known name and is the conventional diagnostic here.

B - Incorrect: Ping primarily tests reachability; displaying a resolved address is not the direct DNS-record query requested.

C - Incorrect: Traceroute inspects responding hops rather than directly querying DNS address records.

D - Incorrect: Netstat reports existing endpoints/connections and is not a direct lookup for an arbitrary server name.

Primary note proof

Study-note section NSA06

> Resolvers cache answers for their TTL. Query nslookup name or specify the DNS server explicitly; compare the configured resolver with the authoritative answer when appropriate. A successful ping by address but failed name lookup suggests naming/configuration trouble. Ping by name may display a resolved address, but nslookup is the direct DNS diagnostic. “Resolving” is a general term that includes forward resolution.

Study-note section NSA07

> netstat displays connections/endpoints/protocols and TCP states; netstat -ano displays numeric addresses/ports and PIDs. Use the exact command name netstat rather than netstart. ping probes IP reachability; traceroute/tracert reveals responding hops using TTL/hop-limit expiry. ICMP filtering can prevent replies without proving the service is down. nslookup inspects DNS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 27 | lines 26-28

> Ping (Packet Internet Groper) uses ICMP echo
> messages to check the physical and logical
> connectivity of machineson a network.

Data Communication and Computer Networking module | PDF page 27 | lines 29-31

> Traceroute Using ICMP timeouts, Traceroute is
> used to discover the path a packet takes as it
> traverses an Internetwork.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

> /etc/hosts
> The /etc/hosts file always contains the localhost IP
> address, 127.0.0.1, which is
> usedfor interprocess communication.
> Never remove this line! Sometimes contains
> addresses of additional hosts, whichcan be contacted
> without using an external naming service such as
> DNS (the Domain Name Server).

Question sources

Computer Science model exam · Question 85

University of Gondar model exam · Question 84

Question 11 | Q00355

LAN clients must reach public IPv4 destinations through a gateway with working routes. Which listed address situation commonly calls for source NAT at that gateway?

A. Default gateway of LAN isn't configured

B. Global IP address is used in the LAN

C. He/she don't know type of IP address used in the LAN

D. Private IP address is used in the LAN

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: Private IP address is used in the LAN

Private IPv4 addresses are not directly routed as public Internet destinations. A gateway can translate those source addresses, often using port bindings so several clients share a public address. This works together with valid routing and the external connection. A missing default gateway must be repaired separately, and public addressing or uncertainty about the current configuration is not by itself a reason to enable NAT.

Why each choice is right or wrong

A - Incorrect: A missing gateway requires routing configuration; address translation is not a replacement for it.

B - Incorrect: Public addressing in a LAN does not automatically require address translation.

C - Incorrect: Not knowing the address type calls for inspecting configuration, not blindly enabling NAT.

D - Correct: Private IPv4 clients commonly need source translation when reaching public destinations through this gateway.

Primary note proof

Study-note section NSA03

An IPv4 address is 32 bits; a CIDR prefix determines its network. Compute network=address AND mask; devices use a route/gateway for remote subnets. Private ranges are 10/8,172.16/12 and 192.168/16; private addresses are not directly Internet-routable. Neither IP alone nor a class label determines today's subnet.

Study-note section NSA11

Routing selects a next hop using a destination prefix; NAT rewrites addresses. Many-to-one address-and-port translation (PAT/NAPT) maps internal flows to one public address using distinct port bindings. For example, two private client sockets can appear as public-address ports 40001 and 40002, with replies translated back to the correct clients.

Study-note section NSA11

NAT is often used when private IPv4 clients access public networks; it does not itself replace a gateway, route, firewall or authentication. One-to-one NAT and destination translation are other forms, so NAT does not always mean many hosts share one address. Public addressing inside a LAN does not automatically require NAT. Unsolicited inbound service access may require explicit mapping and filtering.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Question sources

Computer Science model exam · Question 86

University of Gondar model exam · Question 85

Question 12 | Q00356

Which single option gives the complete set of scenarios where DHCP can meet the stated address-management goal, assuming a correctly deployed service?

A. A small network needing automatic host address configuration.

B. Fixed-purpose devices needing predictable addresses through DHCP reservations.

C. A large device population needing centrally managed address configuration.

D. All are correct

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: All are correct

DHCP can reduce repetitive configuration in small as well as large networks. Reservations allow a known device to receive a predictable address while retaining centrally supplied options. A complex population of clients may also benefit from centrally managed scopes and configuration. The scenarios state these goals rather than treating size or complexity alone as a mandate. Each individual option is valid but incomplete as the requested complete set.

Why each choice is right or wrong

A - Incorrect: DHCP can automate this small network, but A alone omits the other qualifying scenarios.

B - Incorrect: Reservations meet the predictable-address goal, but B alone is incomplete.

C - Incorrect: Centralized DHCP management can meet this goal, but C alone is incomplete.

D - Correct: All three stated goals can be met by an appropriately configured DHCP service.

Primary note proof

Study-note section NSA04

> A client renews before expiry; it later tries rebinding if renewal fails. A disconnected host does not automatically renew forever. On lease expiry the address becomes eligible for reuse, subject to server policy. Reconnection normally triggers validation/renewal/new allocation, so an expired lease alone does not prove persistent failure. A reservation gives a predictable address to a known client while retaining DHCP configuration.

Study-note section NSA05

> Use a scope matching the subnet, exclude manually assigned server/router addresses, set correct options and provide a relay when client broadcasts must cross a router. Static assignments and DHCP can coexist; avoid address conflicts. DHCP lowers repetitive administration but availability/security depend on deployment and protection from rogue replies. It does not guarantee connectivity whenever a device changes location.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

> The Process/Application Layer Protocols
> • Telnet
> • File Transfer Protocol (FTP)
> • Network File System (NFS).
> • Simple Mail Transfer Protocol (SMTP
> • POP3 is used to receive mail.
> • Simple Network Management Protocol (SNMP)
> • Domain Name Service (DNS)
> • Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science model exam · Question 87

University of Gondar model exam · Question 86

Question 13 | Q00550

A disconnected client returns after its old DHCP lease expired. The correct scope has 20 assignable addresses, all now leased to other active clients; the server and relay are reachable. What best explains why the returning client cannot obtain an address?

A. the IP address lease has expired on the DHCP server.

B. the DHCP scope needs to be redefined.

C. the DHCP pool limit has been reached.

D. the DHCP server needs to be restarted.

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C: the DHCP pool limit has been reached.

A reachable DHCP service still needs an available address in the correct pool. Here all twenty assignable addresses are already leased to other active clients, so the returning client cannot receive a free dynamic allocation. Its expired lease permits reuse but does not alone explain persistent failure after reconnection. The pool-capacity observation supplies the diagnosis; it does not establish a need to redefine the scope or restart the server.

Why each choice is right or wrong

A - Incorrect: The old lease expired, but normal reacquisition would succeed if pool capacity remained; expiry alone is insufficient.

B - Incorrect: The scope is correct; the evidence identifies exhausted capacity rather than an incorrect subnet definition.

C - Correct: All assignable pool addresses are leased, so the pool limit prevents a new allocation.

D - Incorrect: The server is reachable and processing requests; a restart does not create free pool capacity.

Primary note proof

Study-note section NSA04

A client renews before expiry; it later tries rebinding if renewal fails. A disconnected host does not automatically renew forever. On lease expiry the address becomes eligible for reuse, subject to server policy. Reconnection normally triggers validation/renewal/new allocation, so an expired lease alone does not prove persistent failure. A reservation gives a predictable address to a known client while retaining DHCP configuration.

Study-note section NSA05

For a returning client, inspect current lease/address, reachable server/relay, free pool capacity and logs. New devices can exhaust a pool, but absent observations this is only a hypothesis. Some clients self-configure IPv4 link-local addresses such as 169.254.2.33 after DHCP failure; these support same-link communication, not ordinary routed Internet access. 0.0.0.0 is unspecified and 255.255.255.255 is limited broadcast.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science model exam · 2015 · Question 19

Question 14 | Q00563

After a NIC replacement, the adapter is recognized and enabled in firmware and Device Manager, but there is no link light. On the same switch port, replacing only the CAT5 cable with a known-good cable restores connectivity. Which listed fault is isolated?

A. the CAT5 cable is defective

B. the NIC is disabled in BIOS

C. The NIC is disabled in Device Manager

D. The RJ11 jack is defective

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: the CAT5 cable is defective

The test changes one relevant factor while keeping the adapter and switch port unchanged. Connectivity returns with a known-good cable, which isolates the original cable or its terminations as the failing component. The explicit enabled-state checks exclude the two disabling explanations. RJ11 is a telephone connector rather than the Ethernet connection being tested, so it does not describe the observed fault.

Why each choice is right or wrong

A - Correct: Replacing only the cable restores the link, isolating the original CAT5 cable or its terminations.

B - Incorrect: The adapter is explicitly enabled in firmware.

C - Incorrect: The adapter is explicitly enabled in Device Manager.

D - Incorrect: RJ11 is a telephone connector, not the expected Ethernet RJ45 connection in this NIC scenario.

Primary note proof

Study-note section NSA09

> Check power, cable/connector and link state, then NIC/driver state, VLAN membership, address/prefix, local gateway, route, DNS and finally application port/authentication. A recognized NIC makes a missing driver less likely, not proof of a working physical link. Ethernet commonly uses RJ45 connectors; an RJ11 telephone jack is not the expected Ethernet connection.

Study-note section NSA09

> If same-subnet traffic works but remote traffic fails, compare gateway/routes and filtering. If ping fails but the application connection works, ICMP filtering is plausible. If an application port is refused, the destination is reachable enough to reject it; inspect service/listener policy. A timeout can result from loss, filtering or a down host and needs more evidence.

Study-note section NSA08

Define the symptom, scope, time and recent changes; gather the configuration, baseline and logs. Form a testable hypothesis, test the least disruptive explanation, isolate the failing component, apply a targeted fix, verify the user service and document the result. Change one relevant factor at a time when possible. A reboot can hide evidence and is not a diagnosis by itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 16-18

• Diagnosing and troubleshooting problems on the network and determining the best course of action to take to solve them.

Question sources

Computer Science model exam · 2015 · Question 32

Question 15 | Q00586

Permissions of a file are represented by which of the following characters?

A. r,w,x

B. w,x

C. x,w,e

D. x,w

Hint

Identify the platform and permission model; for an rwx string, separate owner/group/other triplets and use the stated bit weights.

Answer A: r, w and x

NSA19 groups ordinary Unix permission bits into owner/group/other triplets, each containing read, write and execute. Their conventional letters are r, w and x, with numeric weights 4, 2 and 1 respectively. Only A contains the complete set; the alternatives omit read or substitute a nonpermission letter.

Why each choice is right or wrong

A - Correct: r, w and x denote the complete read, write and execute permission set.

B - Incorrect: This omits r, the read bit, and therefore is not the full ordinary permission-letter set.

C - Incorrect: The letter e is not an ordinary Unix rwx permission bit and read is missing.

D - Incorrect: This lists execute and write but omits the read permission represented by r.

Primary note proof

Study-note section NSA19

The nine ordinary permission bits are owner, group and other triplets. In each triplet read=4,write=2,execute=1; sum enabled bits to get an octal digit. A leading ls -l character denotes file type, not another permission bit. For directories, read lists names, execute permits traversal/search, and write changes entries subject to other checks. File deletion usually depends on parent directory permissions, with sticky-bit/ACL rules also possible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 43 | lines 19-35

In Windows, a user account or a user group can receive one of the following permissions to any file or folder:
Read: allows the viewing and listing of a file or folder. When viewing a folder, you can view all its files and subfolders.
Write: allows writing to a file or adding files and subfolders to a folder.
List folder contents: this permission can be assigned only to folders. It permits the viewing and listing of files and subfolders, as well as executing files that are found inthat folder.
Read & execute: permits the reading and accessing of a file‘s contents as well as its execution. When dealing withfolders, it allows the viewing and listing of files and subfolders, as well as the execution of files.

Data Communication and Computer Networking module | PDF page 63 | lines 8-16

Sorts of Files
Most files are just files, called regular files; they contain normal data, for example text files, executable files or programs, input for or output from a program and so on. The -l option to ls displays the file type, using the first character of each input line:
The following table gives an overview of the characters determining the file type:

Question sources

Computer Science model exam · 2015 · Question 55

Question 16 | Q00633

In a centralized client-server deployment, where must centralized file/directory service software be installed at minimum? Client networking support is a separate component.

A. Server

B. every system in the network

C. both server and every system in the network

D. None

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: Server

The question distinguishes server-side service software from the networking support required on every participant. A centralized service runs on a server that provides it; clients need their own networking support but do not all need the server component installed. Options requiring deployment on every system therefore exceed the minimum location asked for. This does not imply that client operating systems lack networking capabilities.

Why each choice is right or wrong

A - Correct: The server hosts the centralized server-side component.

B - Incorrect: Installing the server component on every system is not the minimum required deployment.

C - Incorrect: Requiring every client as well as the server exceeds the minimum server-side placement.

D - Incorrect: A centralized service does require a host, so none is false.

Primary note proof

Study-note section NSA02

> Clients request services; servers provide them. Peer-to-peer systems can do both. A server can also be a client of DNS or backups, so dedicated role does not mean it never makes requests. A network OS is not universally installed only on a dedicated server: clients also require networking support. Specify centralized service software versus the OS on all participants.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 32 | lines 23-30

> Windows workgroup
> A Windows workgroup is a group of standalone
> computers in a peer-to-peer network. Each computer in
> the workgroup uses its own local accounts database to

authenticate resource access. The computers in a workgroup also do not have a common authentication process. The default-networking environment for a clean windows load is workgroup

Data Communication and Computer Networking module | PDF page 34 | lines 7-14

Domain Controllers
A domain controller (DC) is a server computer that responds to security authentication requests within a computer network domain. It is a network server that is responsible for allowing end devices to access shared domain resources. It authenticates users, stores user account information and enforces security policy for a domain. It is most commonly implemented in Microsoft

Question sources

Computer Science model exam · 2015 · Question 102

Question 17 | Q00635

Which listed item is a native Linux filesystem for ordinary user files, rather than a foreign-origin filesystem or paging area?

A. ext2

B. hpfs

C. swap

D. All

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: ext2

Ext2 is a Linux filesystem used for ordinary files. HPFS has its historical origin in OS/2, although Linux can access it when the appropriate driver is available. Swap supplies paging backing space rather than an ordinary user-file filesystem. The native-origin scope therefore identifies ext2 without incorrectly denying Linux support for foreign filesystems or treating swap as a conventional file-storage format.

Why each choice is right or wrong

A - Correct: Ext2 is the native Linux filesystem listed.

B - Incorrect: HPFS is historically associated with OS/2, so it is not the native Linux example requested.

C - Incorrect: Swap is paging backing space, not an ordinary filesystem for storing user files.

D - Incorrect: The all option includes swap as a filesystem, contrary to the note's explicit distinction.

Primary note proof

Study-note section NSA20

ext2 is a Linux filesystem; ext4 is a later filesystem; HPFS is historically associated with OS/2, while swap is backing space rather than an ordinary user-file filesystem. fsck checks and optionally repairs filesystems through filesystem-specific tools; it is not init, halt or an inittab file. Repairs require the filesystem's supported offline/safe procedure. Mount attaches a filesystem to a directory; partitioning, mounting, quotas and capacity belong to storage administration.

Study-note section NSA20

Linux can access some foreign-origin filesystems, including HPFS when its driver is available. Filesystem origin and Linux driver support are different questions. Swap is a paging area rather than an ordinary user-file filesystem. ext2 is a native Linux-filesystem example; broader Linux filesystem support can also include HPFS, so distinguish origin from driver support.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 63 | lines 8-16

Sorts of Files
Most files are just files, called regular files; they
contain normal data, for example text files,
executable files or programs, input for or output
from a program and so on. The -l option to ls
displays the file type, using the first character of
each input line:
The following table gives an overview of the
characters determining the file type:

Data Communication and Computer Networking module | PDF page 64 | lines 8-12

/lost+found: Every partition has a lost+found for files
that were saved during failures are here.
/misc: For miscellaneous purposes.
/mnt: Standard mount point for external file systems,
e.g. a CD-ROM or a digital camera.

Question sources

Computer Science model exam · 2015 · Question 104

Question 18 | Q00650

The top-level directory on a Linux system is represented as _____

A. /

B. $

C. *

D. #

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: /

PDF p.63 states that the tree begins at a forward slash and calls that directory root. Thus the top-level directory is /.

Why each choice is right or wrong

A - Correct: The forward slash is explicitly the root directory (p.63).

B - Incorrect: The cited root symbol is /, not $.

C - Incorrect: The cited root symbol is /, not *.

D - Incorrect: The cited root symbol is /, not #.

Primary note proof

Study-note section NSA20

> / is the filesystem root; /root is the administrative user’s home. /etc stores configuration, /home ordinary user homes, /var variable data/logs, /dev device nodes, /proc kernel/process information, /boot boot material and /tmp temporary files. Cleanup/mount layouts vary; do not assume every partition has lost+found or every reboot erases every temporary directory.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 63 | lines 19-22

> The tree of the file system starts at the trunk or slash,
> indicated by a forward slash (/). This directory,
> containingall underlying directories and files, is also
> called root directory or —the root‖ of the file system.

Question sources

Computer Science model exam · 2015 · Question 119

Question 19 | Q00900

Which listed item is a traffic-prioritization mechanism that can operate across layers, rather than an application mediator or service installation/configuration activity?

A. Proxies and agents

B. Installing a new service

C. .
E-mail configuration

D. Quality of service

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: Quality of service

Quality of service classifies, queues and prioritizes traffic to meet performance objectives such as low delay or limited loss. It can span layers while supporting applications. Proxies and agents are mediating components, and installing a service or configuring email are administrative activities. The revised question uses those explicit functional categories, so it does not require an unsupported taxonomy that labels every item an application-layer service.

Why each choice is right or wrong

A - Incorrect: Proxies and agents mediate requests as components rather than naming the traffic-prioritization mechanism.

B - Incorrect: Installing a service is an administrative activity, not a traffic-prioritization mechanism.

C - Incorrect: Email configuration is an administrative activity rather than cross-layer traffic prioritization.

D - Correct: QoS classifies and prioritizes traffic and can operate across layers.

Primary note proof

Study-note section NSA12

> A proxy acts on behalf of a client or backend service and can cache, filter or mediate application requests. A default IP gateway is a next-hop router; an application gateway may translate protocols. It need not be a modem/router combination. These terms describe functions that can coexist on one machine.

Study-note section NSA12

> QoS classifies, queues and prioritizes traffic to meet performance objectives such as low delay or limited loss. It can operate across layers and support applications, so it is not exclusively an application-layer service. Installing a new service is an administrative action; email configuration is another action and proxies/agents are components. Distinguish service-management actions from components using a stated taxonomy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 48 | lines 41-47

Proxies and Gateways
What is proxy server?
A proxy server acts as a gateway between you and the Internet. It‘s an intermediary server separating end users from the websites they browse. Proxy servers provide varying levels of functionality, security, and privacy depending on your use case, needs, or company policy.

Question sources

Computer Science Exit Exam · 2018 · Question 13

Question 20 | Q00924

A data-center volume has exactly two equal disks and must retain service after either disk fails. One disk’s usable capacity is sufficient. Which listed RAID level meets these requirements?

A. RAID level 3

B. RAID level 1

C. .

RAID level 2

D. RAID level 0

Hint

Compare striping granularity, redundancy placement, usable capacity and required disk-failure tolerance.

Answer B: RAID level 1

With exactly two equal disks, RAID1 mirrors the data and leaves one disk's usable capacity while tolerating either disk failing. RAID0 provides no disk-failure tolerance. RAID2 and RAID3 rely on different striping and redundancy layouts rather than the two-disk mirrored arrangement that meets these requirements. The choice follows the stated disk count, capacity and failure goal rather than the institution's size or industry.

Why each choice is right or wrong

A - Incorrect: RAID3 uses byte striping with dedicated parity, not the required two-disk fault-tolerant layout.

B - Correct: A two-disk RAID1 mirror has one disk's capacity and survives either disk failing.

C - Incorrect: RAID2 uses bit striping with Hamming redundancy, rather than the two-disk solution meeting this requirement.

D - Incorrect: RAID0 has no disk-failure tolerance.

Primary note proof

Study-note section NSA25

> RAID0 stripes for capacity/throughput with no disk-failure tolerance. RAID1 mirrors; a two-disk mirror has one disk's usable capacity and survives one disk failure. RAID2 uses bit striping/Hamming-code redundancy, RAID3 byte striping with dedicated parity, RAID4 block striping with dedicated parity and RAID5 block striping with distributed parity. RAID6 distributes dual parity and tolerates two disk failures. RAID10 stripes mirrored pairs; survival depends on which disks fail.

Study-note section NSA25

> With N equal disks of capacity D, RAID0 usable=ND, paired RAID1/10≈ND/2, RAID5=(N−1)D and RAID6=(N−2)D. Hardware RAID may offload work to controller resources/cache but does not universally outperform software RAID. Rebuilds consume bandwidth; parity writes have overhead. A large university/bank alone does not select a level. Striping plus parity is not uniquely RAID5 unless distributed block parity is specified.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 59 | lines 14-24

> Data is distributed across the drives in one of several ways, referred to as RAID levels, depending on the required level of redundancy and performance. The different schemes, or data distribution layouts, are namedby the word —RAID‖ followed by a number, for example RAID 0 or RAID 1. Each scheme, or RAID level, provides a different balance among the key goals: reliability, availability, performance, and capacity. RAIDlevels greater than RAID 0 provide

protection against unrecoverable sector read errors,
as well as against failures of whole physical drives.

Data Communication and Computer Networking module | PDF page 59 | lines 46-52

RAID 1 consists of data mirroring, without parity or
striping . Data is written identically to two or more
drives,thereby producing a —mirrored set‖ of drives.
Thus, any read request can be serviced by any drive in
the set. If a request is broadcast to every drive in the set,
it can be serviced by the drive that accesses the data first
(depending on its seek time and rotational latency),

Data Communication and Computer Networking module | PDF page 60 | lines 12-26

RAID 2 consists of bit-level striping with dedicated
Hamming-code parity. All disk spindle rotation is
synchronized and data is striped such that each sequential
bit is on a different drive. Hamming-code parity is
calculated across corresponding bits and stored on at least
one parity drive. This level is of historical significance
only; as of 2014 it is not used by any commercially
available system.
RAID 3 consists of byte-level striping with dedicated
parity. All disk spindle rotation is synchronized and data
is striped such that each sequential byte is on a different
drive. Parity is calculated across corresponding bytes and
stored on a dedicated parity drive. Although
implementations exist, RAID 3 is not commonly used in
practice.

Data Communication and Computer Networking module | PDF page 59 | lines 33-45

RAID 0 consists of striping, but no mirroring or
parity. Compared to a spanned volume, the capacity
of a RAID 0 volume is the same; it is the sum of the
capacities of the drives in the set. But because
striping distributes the contents of each file among all
drives in the set, the failure of any drive causes the
entire RAID 0 volume and all files to be lost. In
comparison, a spanned volume preserves thefiles on
the unfailing drives. The benefits of RAID 0 is that
the throughput of read and write operations to any file
is multiplied by the number of drives because, unlike
spanned volumes, reads and writes are done
concurrently.

Question sources

Computer Science Exit Exam · 2018 · Question 37

Question 21 | Q00937

Which listed RAID layout uses block-level striping with distributed parity and tolerates one disk failure?

A. RAID 2

B. RAID 1

C. RAID 5

D. RAID 0

Hint

Compare striping granularity, redundancy placement, usable capacity and required disk-failure tolerance.

Answer C: RAID 5

RAID5 distributes parity across disks while striping data at block level. That distinguishes it from RAID2's bit striping and Hamming-code redundancy. RAID1 mirrors copies instead of organizing the array as distributed-parity striping, while RAID0 adds no redundancy. The explicit granularity and parity placement identify the intended layout rather than treating every form of redundant striping as equivalent.

Why each choice is right or wrong

A - Incorrect: RAID2 uses bit striping with Hamming redundancy, not distributed block parity.

B - Incorrect: RAID1 mirrors data rather than using the specified striping-with-parity organization.

C - Correct: RAID5 uses block striping with distributed parity and one-disk failure tolerance.

D - Incorrect: RAID0 stripes without redundant parity and provides no disk-failure tolerance.

Primary note proof

Study-note section NSA25

> RAID0 stripes for capacity/throughput with no disk-failure tolerance. RAID1 mirrors; a two-disk mirror has one disk's usable capacity and survives one disk failure. RAID2 uses bit striping/Hamming-code redundancy, RAID3 byte striping with dedicated parity, RAID4 block striping with dedicated parity and RAID5 block striping with distributed parity. RAID6 distributes dual parity and tolerates two disk failures. RAID10 stripes mirrored pairs; survival depends on which disks fail.

Study-note section NSA25

> With N equal disks of capacity D, RAID0 usable=ND, paired RAID1/10≈ND/2, RAID5=(N−1)D and RAID6=(N−2)D. Hardware RAID may offload work to controller resources/cache but does not universally outperform software RAID. Rebuilds consume bandwidth; parity writes have overhead. A large university/bank alone does not select a level. Striping plus parity is not uniquely RAID5 unless distributed block parity is specified.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 60 | lines 12-26

RAID 2 consists of bit-level striping with dedicated Hamming-code parity. All disk spindle rotation is synchronized and data is striped such that each sequential bit is on a different drive. Hamming-code parity is calculated across corresponding bits and stored on at least one parity drive. This level is of historical significance only; as of 2014 it is not used by any commercially available system.
RAID 3 consists of byte-level striping with dedicated parity. All disk spindle rotation is synchronized and data is striped such that each sequential byte is on a different drive. Parity is calculated across corresponding bytes and stored on a dedicated parity drive. Although implementations exist, RAID 3 is not commonly used in practice.

Data Communication and Computer Networking module | PDF page 60 | lines 35-46

RAID 5 consists of block-level striping with distributed parity. Unlike RAID 4, parity information is distributed among the drives, requiring all drives but one to be present to operate. Upon failure of a single drive, subsequent reads can be calculated from the distributed parity such that no data is lost. RAID 5 requires at least three disks. Like all single-parity concepts, large RAID 5 implementations are susceptible to system failures because of trends regarding array rebuild time and the chance of drive failure during rebuild. Rebuilding an array requires reading all data from all disks, opening a chance for a second drive failure and the loss of the entire array.

Data Communication and Computer Networking module | PDF page 59 | lines 46-52

RAID 1 consists of data mirroring, without parity or striping . Data is written identically to two or more drives,thereby producing a —mirrored set‖ of drives. Thus, any read request can be serviced by any drive in the set. If a request is broadcast to every drive in the set, it can be serviced by the drive that accesses the data first (depending on its seek time and rotational latency),

Data Communication and Computer Networking module | PDF page 59 | lines 33-45

RAID 0 consists of striping, but no mirroring or parity. Compared to a spanned volume, the capacity of a RAID 0 volume is the same; it is the sum of the

capacities of the drives in the set. But because striping distributes the contents of each file among all drives in the set, the failure of any drive causes the entire RAID 0 volume and all files to be lost. In comparison, a spanned volume preserves thefiles on the unfailing drives. The benefits of RAID 0 is that the throughput of read and write operations to any file is multiplied by the number of drives because, unlike spanned volumes, reads and writes are done concurrently.

Question sources

Computer Science Exit Exam · 2018 · Question 50

Question 22 | Q00954

Which tool directly sends ICMP echo probes to test IP reachability?

A. Ipconfig

B. Ping

C. Encryption

D. Firewall

Hint

Match the requested observation to the tool’s actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer B: Ping

Ping sends ICMP echo requests and observes replies to test IP reachability. A reply confirms that the path and target support that exchange. A missing reply may reflect filtering or loss, so it does not alone prove disconnection. Interface configuration inspection, confidentiality protection and traffic-policy enforcement serve different purposes from sending this diagnostic probe.

Why each choice is right or wrong

A - Incorrect: Ipconfig displays local adapter configuration; it does not send the requested echo probes.

B - Correct: Ping sends ICMP echo requests and checks for replies.

C - Incorrect: Encryption protects confidentiality rather than generating an echo reachability test.

D - Incorrect: A firewall permits or blocks traffic according to policy rather than being the echo-probe utility.

Primary note proof

Study-note section NSA07

Windows ipconfig shows addresses, mask and gateway; ipconfig /all adds complete adapter configuration including physical/MAC address, DHCP and DNS information. /renew requests renewal, /release releases a dynamic address and /flushdns clears the client DNS cache. Linux ip addr and ip route show interface addresses and routes; exact persistent configuration varies by distribution.

Study-note section NSA-S04

Ping uses ICMP echo messages to test IP reachability. A reply shows that the addressed host and path support that exchange; no reply alone does not prove the host is disconnected, because filtering or loss can suppress responses. Traceroute instead discovers responding intermediate hops through packet-lifetime expiry; it is a path-discovery diagnostic, not the same question as an end-to-end echo test.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 27 | lines 26-28

Ping (Packet Internet Groper) uses ICMP echo
messages to check the physical and logical
connectivity of machineson a network.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Question sources

Computer Science Exit Exam · 2018 · Question 67

Question 23 | Q01003

Suppose you are network administrator, you want to determine the path between two connections, often a connection to another device will have to go through multiple routers. Which command would you use?

A. ping

B. nslookup

C. traceroute

D. All of the above

Hint

Match the requested observation to the tool's actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer C: traceroute

Traceroute discovers responding intermediate hops by using packet-lifetime expiry. That provides evidence about the multi-router path toward a destination. Ping instead tests an end-to-end echo exchange, and nslookup asks DNS questions. Neither produces the requested hop sequence, so combining all the tools is not the correct description of the specific path-discovery command. Some routers may suppress responses, leaving gaps in the trace.

Why each choice is right or wrong

A - Incorrect: Ping probes end-to-end reachability, not the intermediate-hop sequence.

B - Incorrect: Nslookup inspects DNS name-resolution information, not router hops.

C - Correct: Traceroute discovers responding intermediate hops toward the destination.

D - Incorrect: Not all listed commands perform hop discovery.

Primary note proof

Study-note section NSA07

> netstat displays connections/endpoints/protocols and TCP states; netstat -ano displays numeric addresses/ports and PIDs. Use the exact command name netstat rather than netstart. ping probes IP reachability; traceroute/tracert reveals responding hops using TTL/hop-limit expiry. ICMP filtering can prevent replies without proving the service is down. nslookup inspects DNS.

Study-note section NSA06

> Resolvers cache answers for their TTL. Query nslookup name or specify the DNS server explicitly; compare the configured resolver with the authoritative answer when appropriate. A successful ping by address but failed name lookup suggests naming/configuration trouble. Ping by name may display a resolved address, but nslookup is the direct DNS diagnostic. "Resolving" is a general term that includes forward resolution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 27 | lines 26-28

> Ping (Packet Internet Groper) uses ICMP echo messages to check the physical and logical connectivity of machineson a network.

Data Communication and Computer Networking module | PDF page 27 | lines 29-31

> Traceroute Using ICMP timeouts, Traceroute is used to discover the path a packet takes as it traverses an Internetwork.

Question sources

National Exit Exam practice questions · Question 17

Question 24 | Q01191

In a deployment with one unprotected DHCP server, which statements are not actual disadvantages of DHCP? Select all that apply.

A. Fault on the DHCP server can affect the whole network

B. It is not a secured way of assigning IP addresses

C. Static IP address assignment can't be used.

D. It minimizes the load of network admin

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C, D: Static IP address assignment can't be used.; It minimizes the load of network admin

A single DHCP service can become an availability dependency for clients needing configuration, and rogue replies are a risk in an unprotected deployment. However, DHCP does not prohibit static assignments: administrators can exclude manually used addresses to avoid conflicts. Reducing repetitive administration is also an advantage. Therefore the false limitation about static addressing and the genuine administrative benefit are the requested exceptions.

Why each choice is right or wrong

A - Incorrect: Depending on deployment, DHCP-server failure can impair clients needing leases, so this can be a disadvantage.

B - Incorrect: Rogue replies and insecure deployment are genuine risks, although DHCP security depends on protections.

C - Correct: This alleged disadvantage is false: static assignments and DHCP can coexist safely with exclusions.

D - Correct: Reducing repetitive administrator work is an advantage rather than a disadvantage.

Primary note proof

Study-note section NSA05

> Use a scope matching the subnet, exclude manually assigned server/router addresses, set correct options and provide a relay when client broadcasts must cross a router. Static assignments and DHCP can coexist; avoid address conflicts. DHCP lowers repetitive administration but availability/security depend on deployment and protection from rogue replies. It does not guarantee connectivity whenever a device changes location.

Study-note section NSA04

> DHCP supplies addresses and configuration options. Typical initial IPv4 allocation is Discover→Offer→Request→ACK. The server manages its scope/pool, allocated leases, exclusions and reservations; options may include mask, gateway and DNS servers. A dynamic lease lasts for a specified interval, not forever.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

> The Process/Application Layer Protocols
> • Telnet
> • File Transfer Protocol (FTP)
> • Network File System (NFS).
> • Simple Mail Transfer Protocol (SMTP
> • POP3 is used to receive mail.
> • Simple Network Management Protocol (SNMP)
> • Domain Name Service (DNS)
> • Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science model exam 2 · Question 12

Question 25 | Q01210

Which listed address is the conventional IPv4 loopback address normally chosen to test the local IP stack? Select one.

A. 127.0.0.1

B. 1.0.0.127

C. 127.0.0.0

D. 127.0.0.255

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: 127.0.0.1

The whole 127/8 range is reserved for loopback, but 127.0.0.1 is the conventional address normally used for this test. The other 127-prefixed choices belong to the broader reserved block without being the conventional address requested. The reversed-looking 1.0.0.127 is outside loopback space. A successful loopback probe tests local protocol processing and does not establish that cables or remote devices work.

Why each choice is right or wrong

A - Correct: 127.0.0.1 is the conventional IPv4 loopback address explicitly named in the note.

B - Incorrect: 1.0.0.127 is outside the reserved 127/8 loopback block.

C - Incorrect: This belongs to loopback-reserved 127/8, but is not the conventional address requested.

D - Incorrect: This also lies in 127/8, but is not the conventional loopback address requested.

Primary note proof

Study-note section NSA03

> 127.0.0.1 conventionally names the local IPv4 loopback interface; 127/8 is reserved for loopback. Loopback tests local protocol processing, not cable, switch or remote reachability. Programs can use it for local services. Several loopback descriptions or several 127/8 addresses may be valid; compare each with the stated role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

> /etc/hosts
> The /etc/hosts file always contains the localhost IP
> address, 127.0.0.1, which is
> usedfor interprocess communication.
> Never remove this line! Sometimes contains
> addresses of additional hosts, whichcan be contacted
> without using an external naming service such as
> DNS (the Domain Name Server).

Data Communication and Computer Networking module | PDF page 27 | lines 26-28

> Ping (Packet Internet Groper) uses ICMP echo
> messages to check the physical and logical
> connectivity of machineson a network.

Question sources

Computer Science model exam 2 · Question 31

Question 26 | Q01216

Most of the time, it is recommended to apply a Hardware RAID into your system in terms of system performance. Why? Because Hardware RAID:

A. A physical storage device that is built into a system

B. Can appear as a single hard disk in the system

C. There is no difference between regular hardware and the RAID devices

D. Can use its own resources

Hint

Compare striping granularity, redundancy placement, usable capacity and required disk-failure tolerance.

Answer D: Hardware RAID can use its own controller resources

NSA25 supports controller resources/cache as a reason hardware RAID may offload work from the host. D directly states that relevant mechanism. Appearing as one disk is a presentation property rather than the performance explanation, and neither simply being physical nor denying differences explains offload. The note also rejects any universal claim that hardware RAID always performs better.

Why each choice is right or wrong

A - Incorrect: Being a physical device alone does not explain any RAID performance advantage or computation offload.

B - Incorrect: A single logical-disk presentation does not establish that host processing work is offloaded.

C - Incorrect: Denying differences provides no performance mechanism and ignores dedicated controller resources.

D - Correct: Dedicated controller resources/cache can perform RAID work rather than relying entirely on the host CPU.

Primary note proof

Study-note section NSA25

> With N equal disks of capacity D, RAID0 usable=ND, paired RAID1/10≈ND/2, RAID5=(N−1)D and RAID6=(N−2)D. Hardware RAID may offload work to controller resources/cache but does not universally outperform software RAID. Rebuilds consume bandwidth; parity writes have overhead. A large university/bank alone does not select a level. Striping plus parity is not uniquely RAID5 unless distributed block parity is specified.

Study-note section NSA25

RAID0 stripes for capacity/throughput with no disk-failure tolerance. RAID1 mirrors; a two-disk mirror has one disk's usable capacity and survives one disk failure. RAID2 uses bit striping/Hamming-code redundancy, RAID3 byte striping with dedicated parity, RAID4 block striping with dedicated parity and RAID5 block striping with distributed parity. RAID6 distributes dual parity and tolerates two disk failures. RAID10 stripes mirrored pairs; survival depends on which disks fail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 59 | lines 14-24

Data is distributed across the drives in one of several ways, referred to as RAID levels, depending on the required level of redundancy and performance. The different schemes, or data distribution layouts, are namedby the word —RAID‖ followed by a number, for example RAID 0 or RAID 1. Each scheme, or RAID level, provides a different balance among the key goals: reliability, availability, performance, and capacity. RAIDlevels greater than RAID 0 provide protection against unrecoverable sector read errors, as well as against failures of whole physical drives.

Question sources

Computer Science model exam 2 · Question 37

Question 27 | Q01271

All of the following are the roles of a system admin related to the file system and disk space management EXCEPT?

A. Assigning user ID and User name for system users

B. Determining desk usage and quota

C. Creating a Mounting point

D. Partition disk space

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: Assigning user ID and User name for system users

PDF p.42 assigns account permissions and capabilities to user administration, p.53 monitors free disk space, and p.64 identifies a mount point. Assigning a username/ID is the account task, while the other options concern storage organization or capacity.

Why each choice is right or wrong

A - Correct: This belongs to account administration (p.42), making it the requested exception.

B - Incorrect: Disk usage and quota concern the storage capacity being monitored on p.53.

C - Incorrect: A mounting point belongs to the filesystem organization illustrated by /mnt (p.64).

D - Incorrect: Partitioning concerns organizing disk space, within the storage subject rather than naming users.

Primary note proof

Study-note section NSA20

> ext2 is a Linux filesystem; ext4 is a later filesystem; HPFS is historically associated with OS/2, while swap is backing space rather than an ordinary user-file filesystem. fsck checks and optionally repairs filesystems through filesystem-specific tools; it is not init, halt or an inittab file. Repairs require the filesystem's supported offline/safe procedure. Mount attaches a filesystem to a directory; partitioning, mounting, quotas and capacity belong to storage administration.

Study-note section NSA17

> Create named accounts, assign only needed groups/rights, disable departed users and audit privileged activity. Authentication verifies the account; authorization permits operations. A user right such as remote logon differs from a permission on one file. Group-based administration is easier to review than many hidden individual grants.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 42 | lines 36-49

> All user accounts have specific capabilities, privileges,
> and rights. When you create a user account, you can grant
> the user specific capabilities by making the user a member
> of one or more groups. This gives the user the capabilities
> of these groups. You then assign additional capabilities by
> making a user a member of the appropriate groups or
> withdraw capabilities by removing a user from a group.
> An important part of an administrator‘s job is being able
> to determine and set permissions, privileges, and logon
> rights as necessary. Although you can‘t change a group‘s
> built-in capabilities, you can change a group‘s default
> privileges and logon rights. For example, you could
> revoke network access to a computer by removing agroup‘s
> right to access the computer from the network.

Data Communication and Computer Networking module | PDF page 53 | lines 2-20

> Monitoring storage normally takes place at two
> different levels:
> • Monitoring for sufficient disk space
> • Monitoring for storage-related

performanceproblems
The reason for this is that it is possible to have dire problems in one area and no problems whatsoever in the other. For example, it is possible to cause a disk drive to run out of disk space without once causing any kind of performance-related problems. Likewise, it is possible tohave a disk drive that has 99% free space, yet is being pushed past its limits in terms of performance.
In any case, the following statistics are useful for monitoring storage:
• Free Space
• File System-Related Statistics
• Transfers per Second
• Reads/Writes per Second

Data Communication and Computer Networking module | PDF page 64 | lines 8-12

/lost+found: Every partition has a lost+found for files that were saved during failures are here.
/misc: For miscellaneous purposes.
/mnt: Standard mount point for external file systems, e.g. a CD-ROM or a digital camera.

Question sources

Computer Science model exam 2 · Question 92

Question 28 | Q01307

In the ordinary Unix owner/group/other rwx permission-bit model, which operation has no separate file permission bit?

A. Read

B. Write

C. Execute

D. Delete

Hint

Identify the platform and permission model; for an rwx string, separate owner/group/other triplets and use the stated bit weights.

Answer D: Delete

Each ordinary Unix permission triplet contains read, write and execute bits. There is no fourth delete bit in that model. Deletion generally depends on permissions on the parent directory, with other mechanisms such as sticky bits or ACLs able to add restrictions. Windows can have an explicit delete permission, so naming the Unix rwx model is essential to make the exception accurate.

Why each choice is right or wrong

A - Incorrect: Read is a recognized permission in both Windows and ordinary Unix permission models.

B - Incorrect: Write is a recognized access permission, not the requested generic exception.

C - Incorrect: Execute/read-execute permissions exist in the models described in the note.

D - Correct: Unix rwx has no separate delete bit; deletion usually depends on the parent directory.

Primary note proof

Study-note section NSA19

> The nine ordinary permission bits are owner, group and other triplets. In each triplet read=4,write=2,execute=1; sum enabled bits to get an octal digit. A leading ls -l character denotes file type, not another permission bit. For directories, read lists names, execute permits traversal/search, and write changes entries subject to other checks. File deletion usually depends on parent directory permissions, with sticky-bit/ACL rules also possible.

Study-note section NSA18

> Deleting can be an explicit permission in Windows; specify the operating system before comparing permission names. Modify and full control are not identical: full control includes changing permission/ownership capabilities. Separate data access from logon rights. A user allowed to read a file is not necessarily allowed to write it or administer its parent folder.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 43 | lines 19-35

> In Windows, a user account or a user group can receive one of the following permissions to any file or folder:
> Read: allows the viewing and listing of a file or folder. When viewing a folder, you can view all its files and subfolders.
> Write: allows writing to a file or adding files and subfolders to a folder.
> List folder contents: this permission can be assigned only to folders. It permits the viewing and listing of files and subfolders, as well as executing files that are found inthat folder.

> Read & execute: permits the reading and accessing of a file‘s contents as well as its execution. When dealing withfolders, it allows the viewing and listing of files and subfolders, as well as the execution of files.

Data Communication and Computer Networking module | PDF page 43 | lines 36-44

> Modify: when dealing with files, it allows their reading, writing and deletion. When dealing with folders, it allowsthe reading and writing of files and subfolders, plus the deletion of the folder.
> Full control: it allows reading, writing, changing and deleting of any file and subfolder. Generally, files inheritthe permissions of the folder where they are placed, but users can also define specific permissions that are assigned only to a specific file.

Question sources

Operating Systems practice questions · Question 25

Question 29 | Q01495

After DHCP fails at startup, a client self-configures an IPv4 link-local interface address for same-link communication. Which address could it assign?

A. 255.255.255.255

B. 127.0.0.1

C. 169.254.2.33

D. 0.0.0.0

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C: 169.254.2.33 as an IPv4 link-local fallback

An IPv4 link-local fallback uses the 169.254 range; 169.254.2.33 is a usable example. This can permit communication with peers on the same link, but does not provide ordinary routed Internet connectivity. The other values serve special broadcast, loopback or unspecified-address roles. In particular, 0.0.0.0 may appear before configuration but is not the assigned link-local address requested here.

Why each choice is right or wrong

A - Incorrect: This is the limited-broadcast destination, not an assigned host address.

B - Incorrect: Loopback addresses the local host, not its link-local interface.

C - Correct: 169.254.2.33 is an IPv4 link-local fallback address.

D - Incorrect: The unspecified address is not a configured link-local host address.

Primary note proof

Study-note section NSA05

> For a returning client, inspect current lease/address, reachable server/relay, free pool capacity and logs. New devices can exhaust a pool, but absent observations this is only a hypothesis. Some clients self-configure IPv4 link-local addresses such as 169.254.2.33 after DHCP failure; these support same-link communication, not ordinary routed Internet access. 0.0.0.0 is unspecified and 255.255.255.255 is limited broadcast.

Study-note section NSA03

> 127.0.0.1 conventionally names the local IPv4 loopback interface; 127/8 is reserved for loopback. Loopback tests local protocol processing, not cable, switch or remote reachability. Programs can use it for local services. Several loopback descriptions or several 127/8 addresses may be valid; compare each with the stated role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

> The Process/Application Layer Protocols
> • Telnet
> • File Transfer Protocol (FTP)
> • Network File System (NFS).
> • Simple Mail Transfer Protocol (SMTP
> • POP3 is used to receive mail.
> • Simple Network Management Protocol (SNMP)
> • Domain Name Service (DNS)
> • Dynamic Host Configuration Protocol (DHCP)

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

> /etc/hosts
> The /etc/hosts file always contains the localhost IP
> address, 127.0.0.1, which is
> usedfor interprocess communication.
> Never remove this line! Sometimes contains
> addresses of additional hosts, whichcan be contacted
> without using an external naming service such as
> DNS (the Domain Name Server).

Question sources

Mock exam 1 · Question 11

Question 30 | Q01592

Which action is not sound OS hardening when required services must remain available?

A. Remove unnecessary applications and services

B. Install patches

C. Configure users and permissions

D. Block every port, including ports required by legitimate services.

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: Block every port, including ports required by legitimate services.

Hardening reduces unnecessary exposure while preserving the services the system must provide. Removing unused software, deploying supported patches and restricting accounts or permissions all support that goal. Filtering unused ports can help too, but indiscriminately blocking ports required by legitimate services destroys availability. The port option states that harmful action explicitly rather than treating every form of port filtering as inappropriate.

Why each choice is right or wrong

A - Incorrect: Removing unnecessary applications/services reduces attack surface and is an explicit hardening action.

B - Incorrect: Supported patches repair known weaknesses and are part of hardening.

C - Incorrect: Restricting users and permissions is an explicit hardening measure.

D - Correct: Blocking ports required by legitimate services defeats availability rather than sound hardening.

Primary note proof

Study-note section NSA27

> Hardening reduces exposure through unnecessary-service removal, supported patches, restricted accounts/permissions and appropriate port filtering. Blocking an unused known port can be hardening; blocking every useful port merely destroys availability. Weak passwords and unpatched exploitable software are vulnerabilities; attacks exploit weaknesses, and risk combines likelihood with impact.

Study-note section NSA01

> Use documentation, standardization, least privilege, automation and measured change. Record the current state, intended change, rollback and validation. Schedule maintenance according to service impact and urgency, not merely because a calendar date arrived. Fault management restores failures; performance management measures service behavior; configuration management tracks settings; security management controls access; accounting tracks usage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 19-33

Although the specifics of being a system administrator may change from platform to platform, there are underlying themes that do not. These themes make up the philosophy of system administration. The themes are:
• Automate everything
• Document everything
• Communicate as much as possible
• Know your resources
• Know your users
• Know your business
• Security cannot be an afterthought
• Plan ahead
• Expect the unexpected
• Backup and disaster recovery planning
• Patching

Data Communication and Computer Networking module | PDF page 58 | lines 33-40

Disadvantages of Remote Administration
Remote administration has many disadvantages too apart from its advantages. The first and foremost disadvantage is the security. Generally, certain ports to be open at Server level to do remote administration. Due to open ports, the hackers/attackers takes advantage to compromise the system. It is advised that remote administration to be used only in emergency or essential

Data Communication and Computer Networking module | PDF page 42 | lines 36-49

All user accounts have specific capabilities, privileges, and rights. When you create a user account, you can grant the user specific capabilities by making the user a member of one or more groups. This gives the user the capabilities of these groups. You then assign additional capabilities by making a user a member of the appropriate groups or withdraw capabilities by removing a user from a group. An important part of an administrator‘s job is being able to determine and set permissions, privileges, and logon rights as necessary. Although you can‘t change a group‘s built-in capabilities, you can change a group‘s default privileges and logon rights. For example, you could revoke network access to a computer by removing agroup‘s right to access the computer from the network.

Data Communication and Computer Networking module | PDF page 59 | lines 2-4

situations only to do administration remotely. In normal situations, it is ideal to block the ports to avoid remote administration.

Question sources

Mock exam 3 · Question 24

Question 31 | SM0028

You know the upstairs computer's IP address and want a basic end-to-end ICMP echo reachability test, rather than a router-hop trace. Which command should you use?

A. nslookup

B. ping

C. DHCP discover

D. traceroute

Hint

Match the requested observation to the tool's actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer B: ping

Ping directly tests whether the addressed computer and path support an ICMP echo exchange. A reply is useful reachability evidence; absence of a reply may result from filtering or loss. Nslookup tests DNS, DHCP discovery seeks configuration service, and traceroute identifies responding intermediate hops. These answer different operational questions from the basic echo test requested.

Why each choice is right or wrong

A - Incorrect: Nslookup queries DNS, which is unnecessary for the specified known-IP echo test.

B - Correct: Ping performs the requested end-to-end ICMP echo test.

C - Incorrect: DHCP discovery locates configuration service rather than testing this remote host.

D - Incorrect: Traceroute probes the hop path rather than performing the requested basic echo test.

Primary note proof

Study-note section NSA-S04

Ping uses ICMP echo messages to test IP reachability. A reply shows that the addressed host and path support that exchange; no reply alone does not prove the host is disconnected, because filtering or loss can suppress responses. Traceroute instead discovers responding intermediate hops through packet-lifetime expiry; it is a path-discovery diagnostic, not the same question as an end-to-end echo test.

Study-note section NSA07

netstat displays connections/endpoints/protocols and TCP states; netstat -ano displays numeric addresses/ports and PIDs. Use the exact command name netstat rather than netstart. ping probes IP reachability; traceroute/tracert reveals responding hops using TTL/hop-limit expiry. ICMP filtering can prevent replies without proving the service is down. nslookup inspects DNS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 27 | lines 26-28

Ping (Packet Internet Groper) uses ICMP echo
messages to check the physical and logical
connectivity of machineson a network.

Data Communication and Computer Networking module | PDF page 27 | lines 29-31

Traceroute Using ICMP timeouts, Traceroute is
used to discover the path a packet takes as it
traverses an Internetwork.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

MOE Computer Science Exit Exam · Question 8-3

National Computer Science Exit Exam · Question 28

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 186

Question 32 | SM0084

Which one of the following is correct about human role in system administration?

A. Ethics is not a requirement to manage computers

B. It requires patience, understanding and knowledge

C. It doesn't need organizational skills

D. System administrator is not expected to have skills and confidence

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer B: Patience, understanding and knowledge

NSA01 includes human abilities and responsible conduct in administration: patience, communication, ethics, technical understanding and organization. B agrees with that account. The other statements deny ethics, organization or expected competence, directly contradicting the note's description of the role rather than identifying a valid human requirement.

Why each choice is right or wrong

A - Incorrect: NSA01 explicitly includes ethics among the qualities required for responsible administration.

B - Correct: Patience and technical understanding/knowledge agree directly with the human qualities listed in NSA01.

C - Incorrect: Organization is explicitly important, so denying organizational skills contradicts the note.

D - Incorrect: The role requires technical understanding and competent administration rather than an absence of skills.

Primary note proof

Study-note section NSA01

> System administrators maintain hosts, accounts, storage, software and availability; network administrators maintain connectivity, addressing, routes, network devices and performance. Security administrators coordinate controls and policy. These duties overlap, so account management, testing patches and training users are not exclusive to one job title. Patience, communication, ethics, technical understanding and organization matter alongside command knowledge.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 29 | lines 16-20

> It‘s about hardware, software, user support,
> diagnosis, repair and prevention. System

administrators need to know a bit of everything: the skills are technical, administrative and socio-psychological.

Question sources

MOE Computer Science Exit Exam · Question 24-1

National Computer Science Exit Exam · Question 83

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 187

Question 33 | SM0270

A planned hardware upgrade requires the system to be powered off, and an acceptable maintenance window and rollback are arranged. Which listed circumstance justifies the planned shutdown?

A. Using regular schedules

B. For maintenance and upgrade

C. To check for security alerts

D. When there is a failure

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer B: For maintenance and upgrade

A shutdown is justified when the required procedure needs it and its service impact has been planned. Here the hardware upgrade explicitly requires power-off and an acceptable window has been arranged. A calendar schedule alone is insufficient, checking alerts ordinarily requires no shutdown, and no failure is described. The decision therefore follows the actual maintenance requirement rather than a universal rule that systems should stop periodically.

Why each choice is right or wrong

A - Incorrect: A regular calendar schedule alone does not justify shutting down a service without considering impact and necessity.

B - Correct: The stated maintenance/upgrade procedure requires a planned power-off.

C - Incorrect: Checking alerts does not ordinarily require stopping the system and supplies no shutdown justification here.

D - Incorrect: No failure is described; the shutdown is required for the planned upgrade.

Primary note proof

Study-note section NSA01

> Use documentation, standardization, least privilege, automation and measured change. Record the current state, intended change, rollback and validation. Schedule maintenance according to service impact and urgency, not merely because a calendar date arrived. Fault management restores failures; performance management measures service behavior; configuration management tracks settings; security management controls access; accounting tracks usage.

Study-note section NSA08

> Define the symptom, scope, time and recent changes; gather the configuration, baseline and logs. Form a testable hypothesis, test the least disruptive explanation, isolate the failing component, apply a targeted fix, verify the user service and document the result. Change one relevant factor at a time when possible. A reboot can hide evidence and is not a diagnosis by itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

> System Monitoring Software Essentials
> In order to keep your system fit for purpose, your
> monitoring activities need to cover the following
> priorities:
> • Acceptable delivery speeds
> • Constant availability
> • Preventative maintenance
> • Software version monitoring and patching
> • Intrusion detection
> • Data integrity
> • Security monitoring
> • Attack mitigation
> • Virus prevention and detection

Question sources

Computer Science Exit Exam · Sene 2016 · Question 23-1

Computer Science Exit Exam practice collection · Question 192

Question 34 | SM0287

Which one of the following is NOT a reason for the need of backups?

A. Security incidents

B. Upgrades and migrations

C. Hardware failures

D. Authorization

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: Authorization

Backups preserve recoverable earlier data after hardware loss, compromise, corruption or failed upgrades and migrations. These events can destroy or damage the live copy, so independent protected copies and tested restores address their consequences. Authorization instead decides who may access current resources. It is essential security policy, but is a separate function from preserving copies for recovery.

Why each choice is right or wrong

A - Incorrect: Compromise can damage or delete data, creating a restoration need.

B - Incorrect: Failed upgrades or migrations can require restoring the previous valid data.

C - Incorrect: Hardware failure can destroy the live copy and require recovery.

D - Correct: Authorization governs current access rights; it is not a data-loss recovery reason.

Primary note proof

Study-note section NSA24

> Backups recover previous data after deletion, compromise, corruption, failed upgrades/migrations or hardware loss. Authorization controls current access, a different function. A mirror/replica also copies deletions or corruption and does not replace versioned independent backups. Keep protected copies in separate failure domains, define retention and test restores.

Study-note section NSA17

> Create named accounts, assign only needed groups/rights, disable departed users and audit privileged activity. Authentication verifies the account; authorization permits operations. A user right such as remote logon differs from a permission on one file. Group-based administration is easier to review than many hidden individual grants.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 6-8

> • Backing up the network software and data
> regularly to protect against the failure of network
> or any of its components

Data Communication and Computer Networking module | PDF page 46 | lines 19-22

> that the user profile data is not being automatically backed
> up by the server. Since most users rarely back up their
> computers, if a hard drive fails, any data that is stored
> within local profiles on that machine would be lost.

Data Communication and Computer Networking module | PDF page 42 | lines 36-49

> All user accounts have specific capabilities, privileges, and rights. When you create a user account, you can grant the user specific capabilities by making the user a member of one or more groups. This gives the user the capabilities of these groups. You then assign additional capabilities by making a user a member of the appropriate groups or withdraw capabilities by removing a user from a group. An important part of an administrator‘s job is being able to determine and set permissions, privileges, and logon rights as necessary. Although you can‘t change a group‘s built-in capabilities, you can change a group‘s default privileges and logon rights. For example, you could revoke network access to a computer by removing agroup‘s right to access the computer from the network.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 28-3

Computer Science Exit Exam practice collection · Question 190

Question 35 | SS099

The ordinary file permission string -rwxr--r-- is represented by which octal mode?

A. 744

B. 700

C. 755

D. 766

Hint

Identify the platform and permission model; for an rwx string, separate owner/group/other triplets and use the stated bit weights.

Answer A: 744.

The leading hyphen identifies an ordinary file and is not a permission bit. Split the remaining characters as rwx | r-- | r--. The owner has 4+2+1=7, the group has read only (4), and others have read only (4). Writing those three octal digits in order gives 744, so A is correct.

Why each choice is right or wrong

A - Correct: 744 exactly represents owner rwx, group r-- and others r--.

B - Incorrect: 700 would deny both group and others all permissions, unlike the two read permissions shown.

C - Incorrect: 755 would give group and others execute permission as well as read permission; those execute bits are absent.

D - Incorrect: 766 would give group and others write permission; neither triplet contains w.

Primary note proof

Study-note section NSA19

The nine ordinary permission bits are owner, group and other triplets. In each triplet read=4,write=2,execute=1; sum enabled bits to get an octal digit. A leading ls -l character denotes file type, not another permission bit. For directories, read lists names, execute permits traversal/search, and write changes entries subject to other checks. File deletion usually depends on parent directory permissions, with sticky-bit/ACL rules also possible.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 63 | lines 8-16

Sorts of Files
Most files are just files, called regular files; they contain normal data, for example text files, executable files or programs, input for or output from a program and so on. The -l option to ls displays the file type, using the first character of each input line:
The following table gives an overview of the characters determining the file type:

Data Communication and Computer Networking module | PDF page 43 | lines 19-35

In Windows, a user account or a user group can receive one of the following permissions to any file or folder:
Read: allows the viewing and listing of a file or folder. When viewing a folder, you can view all its files and subfolders.
Write: allows writing to a file or adding files and subfolders to a folder.
List folder contents: this permission can be assigned only to folders. It permits the viewing and listing of files and subfolders, as well as executing files that are found inthat folder.
Read & execute: permits the reading and accessing of a file‘s contents as well as its execution. When dealing withfolders, it allows the viewing and listing of files and subfolders, as well as the execution of files.

Question sources

Computer Science model exam collection · Question 6

Question 36 | AM0564

What is the primary role of a system administrator?

A. Designing computer networks

B. Developing software applications

C. Managing and maintaining computer systems

D. Providing technical support to end-users

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C: Managing and maintaining computer systems

PDF p.28 describes getting computer systems running and keeping them running; p.29 calls system administration the management of computers. C captures that broad primary role rather than one subtask.

Why each choice is right or wrong

A - Incorrect: Network design is one challenge (p.30), narrower than maintaining computer systems.

B - Incorrect: The definition is operational management, not developing applications (p.28).

C - Correct: Managing and maintaining systems directly matches pp.28-29.

D - Incorrect: User support is included (p.29), but is only part of the broader role.

Primary note proof

Study-note section NSA01

> System administrators maintain hosts, accounts, storage, software and availability; network administrators maintain connectivity, addressing, routes, network devices and performance. Security administrators coordinate controls and policy. These duties overlap, so account management, testing patches and training users are not exclusive to one job title. Patience, communication, ethics, technical understanding and organization matter alongside command knowledge.

Study-note section NSA01

> Use documentation, standardization, least privilege, automation and measured change. Record the current state, intended change, rollback and validation. Schedule maintenance according to service impact and urgency, not merely because a calendar date arrived. Fault management restores failures; performance management measures service behavior; configuration management tracks settings; security management controls access; accounting tracks usage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 46-53

> Philosophy of System Administration
> Network and system administration is a branch of engineering that concerns the operational management of human-computer systems. It is about putting together a network of computers (workstations, PCs and supercomputers), getting them running and then keeping them running in spite of the activities of users who tend to cause the systems to fail.

Data Communication and Computer Networking module | PDF page 29 | lines 25-36

> System administration is the term used traditionally by mainframe and Unix engineers to describe the management of computers whether they are coupled by a network or not. To this community, network administration means the management of network infrastructure devices (routers and switches). The world of personal computers (PCs) has no tradition of managingindividual computers and their subsystems, and thus doesnot speak of system administration, per se. To this community, network administration is the management ofPCs in a network. In this material, we shall take the first view, since this is more precise.

Question sources

Computer Science MCQ practice collection · Question 564

Question 37 | AM0565

Under the note’s primary division of administrative responsibilities, which listed task directly concerns ongoing network connectivity, performance and protection?

A. Installing and configuring operating systems

B. Monitoring network performance and security

C. Developing software applications

D. Providing end-user training

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer B: Monitoring network performance and security

Network administrators primarily maintain connectivity, addressing, routes, devices and performance. Monitoring network performance and security directly matches that operational scope. Operating-system installation and user training may be shared administrative duties, but they do not directly describe the ongoing network-monitoring task asked for. Developing software applications has a different primary purpose. The scope selects the core network responsibility without claiming exclusive job boundaries.

Why each choice is right or wrong

A - Incorrect: OS installation can overlap with administrative work, but is not the ongoing network-monitoring task requested.

B - Correct: Monitoring network performance and security directly matches network-administration responsibilities.

C - Incorrect: Developing application software is not a network-administration responsibility established by this note.

D - Incorrect: User training may be shared, but does not directly name monitoring network operation.

Primary note proof

Study-note section NSA01

> System administrators maintain hosts, accounts, storage, software and availability; network administrators maintain connectivity, addressing, routes, network devices and performance. Security administrators coordinate controls and policy. These duties overlap, so account management, testing patches and training users are not exclusive to one job title. Patience, communication, ethics, technical understanding and organization matter alongside command knowledge.

Study-note section NSA22

> Measure a baseline, sample over an interval, compare with workload and correlate symptoms before changing hardware. CPU utilization/run queues identify processing pressure; low free memory alone is weak evidence because cache may be reclaimable. Persistent paging with high latency indicates memory pressure. Storage capacity and I/O latency are separate: a disk can be nearly empty but saturated, or nearly full while lightly loaded.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 29 | lines 44-56

> Scope of Network administration
> The management of a network, usually called
> network administration, consists of procedures and
> services that keep the network running properly. An
> important part of network management entails
> making sure that the network is available (or up and

running as IT professionals say) when employees and managers need it.Other admin activities are:
• Monitoring the network capacity to ensure that alltransmission requirements can be met.
• Adding capacity to the network by increasing band width interconnecting additional modes, or creating and interconnecting additional networks.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 30 | lines 41-51

The challenges of system administration
System administration is not just about installing operating systems. It is about planning and designing an efficient community of computers so that real users will be able to get their jobs done. That means:
• Designing a network which is logical and efficient.
• Deploying large numbers of machines which can be easily upgraded later.
• Deciding what services are needed.
• Planning and implementing adequate security.
• Providing a comfortable environment for users.

Data Communication and Computer Networking module | PDF page 30 | lines 2-5

• Training people to use the network effectively
• Assisting IT professionals in organizational applications that will make good use of the network‘s capabilities.

Question sources

Computer Science MCQ practice collection · Question 565

Question 38 | AM0566

What is the purpose of system administration?

A. Managing software development projects

B. Ensuring the availability and reliability of computer systems

C. Managing user accounts and permissions

D. Designing computer networks

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer B: Ensuring the availability and reliability of computer systems

Keeping systems running (PDF p.28) and seeking predictability as the basis of reliability (p.31) support availability and reliability as the purpose. User permissions are a means of administration (p.42), not its full goal.

Why each choice is right or wrong

A - Incorrect: Software-project management is not the system-operation goal defined on p.28.

B - Correct: Availability follows from keeping systems running; reliability is explicitly tied to predictability (pp.28,31).

C - Incorrect: Account and permission management is a narrower activity (p.42).

D - Incorrect: Design is one planning task, narrower than the operational goal (pp.28,30).

Primary note proof

Study-note section NSA01

> System administrators maintain hosts, accounts, storage, software and availability; network administrators maintain connectivity, addressing, routes, network devices and performance. Security administrators coordinate controls and policy. These duties overlap, so account management, testing patches and training users are not exclusive to one job title. Patience, communication, ethics, technical understanding and organization matter alongside command knowledge.

Study-note section NSA01

> Use documentation, standardization, least privilege, automation and measured change. Record the current state, intended change, rollback and validation. Schedule maintenance according to service impact and urgency, not merely because a calendar date arrived. Fault management restores failures; performance management measures service behavior; configuration management tracks settings; security management controls access; accounting tracks usage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 46-53

Philosophy of System Administration
Network and system administration is a branch of engineering that concerns the operational management of human-computer systems. It is about putting together a network of computers (workstations, PCs and supercomputers), getting them running and then keeping them running in spite of the activities of users who tend to cause the systems to fail.

Data Communication and Computer Networking module | PDF page 31 | lines 30-35

Principle 2 (Predictability)
The highest level aim in system administration is to work towards a predictable system. Predictability has limits. It is the basis of reliability, hence trust and therefore security.

Data Communication and Computer Networking module | PDF page 42 | lines 36-49

All user accounts have specific capabilities, privileges, and rights. When you create a user account, you can grant the user specific capabilities by making the user a member of one or more groups. This gives the user the capabilities of these groups. You then assign additional capabilities by making a user a member of the appropriate groups or withdraw capabilities by removing a user from a group. An important part of an administrator‘s job is being able to determine and set permissions, privileges, and logon rights as necessary. Although you can‘t change a group‘s built-in capabilities, you can change a group‘s default privileges and logon rights. For example, you could revoke network access to a computer by removing agroup‘s right to access the computer from the network.

Question sources

Computer Science MCQ practice collection · Question 566

Question 39 | AM0567

What is a key principle of system administration?

A. Least privilege

B. Agile development

C. Object-oriented programming

D. Cloud computing

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: Least privilege

NSA01 explicitly identifies least privilege among administration principles: give accounts and processes only the access required for their work. Agile development and object-oriented programming are development approaches; cloud computing is a computing/deployment model. They may occur in administered systems but are not the administration access principle asked for.

Why each choice is right or wrong

A - Correct: Least privilege is explicitly named as a system-administration principle in NSA01.

B - Incorrect: Agile development organizes software work rather than specifying the administration access principle in the note.

C - Incorrect: Object-oriented programming structures software; it is not the stated system-administration principle.

D - Incorrect: Cloud computing is a delivery/deployment model, not the access-control principle listed in NSA01.

Primary note proof

Study-note section NSA17

> Create named accounts, assign only needed groups/rights, disable departed users and audit privileged activity. Authentication verifies the account; authorization permits operations. A user right such as remote logon differs from a permission on one file. Group-based administration is easier to review than many hidden individual grants.

Study-note section NSA-S01

> Least privilege limits an account or process to the rights its task needs. Agile development instead organizes frequent delivery and adaptation of software. Object-oriented programming organizes software around objects with state and behavior. Cloud computing supplies on-demand access to a shared pool of configurable computing resources. A development approach or hosting model does not itself define an access-rights principle.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 31 | lines 17-29

> Principle 1 (Policy is the foundation)
> System administration begins with a
> policy - adecision about what we want
> and what should be, in relation to what we
> can afford.
> Policy speaks of what we wish to

accomplish with the system, and what we
are willing to tolerate of behavior within it.
It must refer to both the component parts and
to the environmentwith which the system
interacts. If we cannot secure predictability,
then we cannot expect
long-term conformance with a policy.

Data Communication and Computer Networking module | PDF page 31 | lines 30-35

Principle 2 (Predictability)
The highest level aim in system
administration is to work towards a
predictable system. Predictability has limits.
It is the basis of reliability, hence trust and
therefore security.

Data Communication and Computer Networking module | PDF page 43 | lines 47-52

Assigning User Rights
The most efficient way to assign user rights is to make the
user a member of a group that already has the right. In
some cases, however, you might want a user to have a
particular right but not have all the other rights of the
group. One way to resolve this problem is to give the user

Question sources

Computer Science MCQ practice collection · Question 567

Question 40 | AM0568

Using the note’s primary role division, which role is responsible for host-local accounts together with host storage, software and availability?

A. Network administrator

B. Database administrator

C. Security administrator

D. System administrator

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: System administrator

System administrators maintain hosts, accounts, storage, software and availability. Network administrators focus on connectivity and routes, while security administrators coordinate controls and policy. Database administration concerns a different resource scope. Responsibilities can overlap, but the stated question asks for the primary host-level role associated with the complete stated set of duties rather than implying that no other administrator ever manages accounts.

Why each choice is right or wrong

A - Incorrect: The primary network role concerns connectivity, addressing and routing rather than this host-level bundle.

B - Incorrect: Database-account administration is a narrower resource scope than the host-level duties listed.

C - Incorrect: Security administration coordinates controls and policy rather than the complete host-maintenance bundle.

D - Correct: System administration is assigned hosts, accounts, storage, software and availability.

Primary note proof

Study-note section NSA01

> System administrators maintain hosts, accounts, storage, software and availability; network administrators maintain connectivity, addressing, routes, network devices and performance. Security administrators coordinate controls and policy. These duties overlap, so account management, testing patches and training users are not exclusive to one job title. Patience, communication, ethics, technical understanding and organization matter alongside command knowledge.

Study-note section NSA17

> Create named accounts, assign only needed groups/rights, disable departed users and audit privileged activity. Authentication verifies the account; authorization permits operations. A user right such as remote logon differs from a permission on one file. Group-based administration is easier to review than many hidden individual grants.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 42 | lines 36-49

> All user accounts have specific capabilities, privileges,
> and rights. When you create a user account, you can grant
> the user specific capabilities by making the user a member
> of one or more groups. This gives the user the capabilities
> of these groups. You then assign additional capabilities by
> making a user a member of the appropriate groups or
> withdraw capabilities by removing a user from a group.
> An important part of an administrator‘s job is being able

to determine and set permissions, privileges, and logon rights as necessary. Although you can‘t change a group‘s built-in capabilities, you can change a group‘s default privileges and logon rights. For example, you could revoke network access to a computer by removing agroup‘s right to access the computer from the network.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Question sources

Computer Science MCQ practice collection · Question 568

Question 41 | AM0569

What is the purpose of network administration?

A. Managing computer hardware

B. Managing computer software

C. Managing network infrastructure

D. Managing user accounts

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C: Managing network infrastructure

PDF p.29 distinguishes the management of computers from management of network infrastructure devices, explicitly adopting the latter meaning for network administration. C is the direct match.

Why each choice is right or wrong

A - Incorrect: General hardware management is broader than the infrastructure-device definition on p.29.

B - Incorrect: General software management is not the adopted network-administration definition.

C - Correct: The module defines network administration as managing routers, switches and network infrastructure (p.29).

D - Incorrect: Account management is an administration subtask but not the adopted infrastructure-focused definition.

Primary note proof

Study-note section NSA01

System administrators maintain hosts, accounts, storage, software and availability; network administrators maintain connectivity, addressing, routes, network devices and performance. Security administrators coordinate controls and policy. These duties overlap, so account management, testing patches and training users are not exclusive to one job title. Patience, communication, ethics, technical understanding and organization matter alongside command knowledge.

Study-note section NSA02

Switching normally forwards frames using MAC addresses within a LAN; routing forwards IP packets between networks. TCP supplies an ordered reliable byte stream, while UDP preserves datagrams without that guarantee. Ports distinguish application endpoints, not physical connectors. Draw switches, links, subnets, gateways, service hosts and dependencies in a network diagram. A topology's shape alone does not guarantee security.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 29 | lines 25-36

System administration is the term used traditionally by mainframe and Unix engineers to describe the management of computers whether they are coupled by a network or not. To this community, network administration means the management of network infrastructure devices (routers and switches). The world of personal computers (PCs) has no tradition of managingindividual computers and their subsystems, and thus doesnot speak of system administration, per se. To this community, network administration is the management ofPCs in a network. In this material, we shall take the first view, since this is more precise.

Question sources

Computer Science MCQ practice collection · Question 569

Question 42 | AM0570

What is the role of a security administrator in system administration?

A. Monitoring network performance

B. Managing user accounts and permissions

C. Enforcing security policies and procedures

D. Configuring network devices

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer C: Enforcing security policies and procedures

PDF p.30 requires putting security procedures in place and ensuring they are followed; p.34 says domain controllers enforce security policy. Enforcing policies and procedures is therefore the best statement of the security role, without assuming exclusive staffing boundaries.

Why each choice is right or wrong

A - Incorrect: Performance monitoring is separately discussed as capacity management (p.29).

B - Incorrect: Account permissions contribute to access security but are narrower than enforcing security procedures.

C - Correct: The duty to ensure security procedures are followed is explicit on p.30.

D - Incorrect: Configuring devices is not by itself the policy-enforcement purpose described on p.30.

Primary note proof

Study-note section NSA26

> A security policy states required behavior, scope, accountable roles and enforcement. Begin with assets/service objectives, assess threats and weaknesses, choose controls, define exceptions/incident procedures, approve/communicate, monitor and periodically review. CIA means confidentiality, integrity and availability; neither sensitive-data protection nor availability alone is the universal primary goal.

Study-note section NSA01

> System administrators maintain hosts, accounts, storage, software and availability; network administrators maintain connectivity, addressing, routes, network devices and performance. Security administrators coordinate controls and policy. These duties overlap, so account management, testing patches and training users are not exclusive to one job title. Patience, communication, ethics, technical understanding and organization matter alongside command knowledge.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

> • Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
> • Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 34 | lines 7-14

> Domain Controllers
> A domain controller (DC) is a server computer that responds to security authentication requests within a computer network domain. It is a network server that is responsible for allowing end devices to access shared domain resources. It authenticates users, stores user account information and enforces security policy for a domain. It is most commonly implemented in Microsoft

Question sources

Computer Science MCQ practice collection · Question 570

Question 43 | AM0571

Which concept in system administration emphasizes the importance of regularly backing up data?

A. Redundancy

B. Fault tolerance

C. Disaster recovery

D. Load balancing

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C: Disaster recovery

PDF p.30 explicitly pairs “Backup and disaster recovery planning.” Its backup duty protects against network/component failure. Disaster recovery is therefore the choice directly connected to regular data backup in the module.

Why each choice is right or wrong

A - Incorrect: Redundancy can protect operation but is not the theme explicitly paired with backups on p.30.

B - Incorrect: Fault tolerance is not the module's named backup-planning concept here.

C - Correct: Backup and disaster recovery planning are explicitly paired (p.30).

D - Incorrect: Load balancing is not connected to backup planning in this passage.

Primary note proof

Study-note section NSA-S02

> Disaster recovery planning prepares for restoring systems and data after disruptive loss. Regular protected backups and tested restoration are part of that plan. Continuing operation despite a component failure is a fault-tolerance objective; possessing extra components alone does not establish that older valid data can be recovered.

Study-note section NSA24

> Backups recover previous data after deletion, compromise, corruption, failed upgrades/migrations or hardware loss. Authorization controls current access, a different function. A mirror/replica also copies deletions or corruption and does not replace versioned independent backups. Keep protected copies in separate failure domains, define retention and test restores.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 19-33

> Although the specifics of being a system administrator
> may change from platform to platform, there are
> underlying themes that do not. These themes make up the
> philosophy of system administration. The themes are:
> • Automate everything
> • Document everything
> • Communicate as much as possible
> • Know your resources
> • Know your users
> • Know your business
> • Security cannot be an afterthought
> • Plan ahead
> • Expect the unexpected
> • Backup and disaster recovery planning
> • Patching

Data Communication and Computer Networking module | PDF page 30 | lines 6-8

> • Backing up the network software and data
> regularly to protect against the failure of network
> or any of its components

Question sources

Computer Science MCQ practice collection · Question 571

Question 44 | AM0572

Which skill describes the general process of diagnosing faults and selecting fixes across system-administration tasks, rather than a specialized design, database or cabling task?

A. Graphic design

B. Database query optimization

C. Troubleshooting and problem-solving

D. Network cable installation

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C: Troubleshooting and problem-solving

Troubleshooting and problem-solving cover defining symptoms, gathering evidence, forming testable hypotheses, isolating causes, applying targeted fixes and verifying service. This process applies across many administration duties. Graphic design, query optimization and cable installation may be useful specialized activities, but none names the broad diagnostic process requested. The scope does not imply that administrators never perform those other tasks.

Why each choice is right or wrong

A - Incorrect: Graphic design concerns visual communication rather than the general diagnostic process.

B - Incorrect: Query optimization improves database query behavior, a specialized task.

C - Correct: Troubleshooting and problem-solving name the requested cross-task diagnostic skill.

D - Incorrect: Cable installation is a physical infrastructure task rather than the general reasoning process.

Primary note proof

Study-note section NSA08

> Define the symptom, scope, time and recent changes; gather the configuration, baseline and logs. Form a testable hypothesis, test the least disruptive explanation, isolate the failing component, apply a targeted fix, verify the user service and document the result. Change one relevant factor at a time when possible. A reboot can hide evidence and is not a diagnosis by itself.

Study-note section NSA09

> Check power, cable/connector and link state, then NIC/driver state, VLAN membership, address/prefix, local gateway, route, DNS and finally application port/authentication. A recognized NIC makes a missing driver less likely, not proof of a working physical link. Ethernet commonly uses RJ45 connectors; an RJ11 telephone jack is not the expected Ethernet connection.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 29 | lines 16-20

> It‘s about hardware, software, user support, diagnosis, repair and prevention. System administrators need to know a bit of everything: the skills are technical, administrative and socio-psychological.

Data Communication and Computer Networking module | PDF page 30 | lines 16-18

> • Diagnosing and troubleshooting problems on the network and determining the best course of action to take to solve them.

Data Communication and Computer Networking module | PDF page 31 | lines 7-14

> Some system administrators are responsible for both the hardware of the network and the computers which it connects, i.e. the cables as well as the computers. Some are only responsible for the computers. Either way, an understanding of how data flow from machine to machineis essential as well as an understanding of how each machine affects every other.

Question sources

Computer Science MCQ practice collection · Question 572

Question 45 | AM0573

Which tasks belong to administrator-side software update and patch management, rather than authoring the software patch? Select all that apply.

A. Developing software patches

B. Testing software updates

C. Installing and deploying software updates

D. Analyzing software vulnerabilities

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer B, C, D: Testing software updates; Installing and deploying software updates; Analyzing software vulnerabilities

Patch administration includes assessing relevant weaknesses, testing or staging an update, preparing rollback, deploying it and verifying the result. These steps help choose and safely apply the appropriate repair. Writing the vendor's software patch is a development activity rather than the operational workflow being classified. The multiple-selection instruction retains the complementary assessment, testing and deployment responsibilities instead of forcing an installation-only answer.

Why each choice is right or wrong

A - Incorrect: Writing the software patch is a development responsibility, not the patch-administration workflow stated here.

B - Correct: Testing/staging updates before deployment is an explicit administrator duty.

C - Correct: Deploying and verifying software updates is also an explicit administrator duty.

D - Correct: Assessing known weaknesses helps prioritize relevant patches and is part of the stated vulnerability-management workflow.

Primary note proof

Study-note section NSA27

> Hardening reduces exposure through unnecessary-service removal, supported patches, restricted accounts/permissions and appropriate port filtering. Blocking an unused known port can be hardening; blocking every useful port merely destroys availability. Weak passwords and unpatched exploitable software are vulnerabilities; attacks exploit weaknesses, and risk combines likelihood with impact.

Study-note section NSA27

> Preventive controls restrict harmful actions, detective controls expose them and recovery controls restore service. Account lifecycle, backup procedures, device configuration and traffic analysis can all be security controls under a stated objective. A vulnerability scan checks known weakness patterns, not bandwidth. Test/stage patches, prepare rollback, deploy and verify; testing and installation are both legitimate administrator duties.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 19-33

> Although the specifics of being a system administrator
> may change from platform to platform, there are
> underlying themes that do not. These themes make up the
> philosophy of system administration. The themes are:
> • Automate everything
> • Document everything
> • Communicate as much as possible
> • Know your resources

• Know your users
• Know your business
• Security cannot be an afterthought
• Plan ahead
• Expect the unexpected
• Backup and disaster recovery planning
• Patching

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

System Monitoring Software Essentials
In order to keep your system fit for purpose, your monitoring activities need to cover the following priorities:
• Acceptable delivery speeds
• Constant availability
• Preventative maintenance
• Software version monitoring and patching
• Intrusion detection
• Data integrity
• Security monitoring
• Attack mitigation
• Virus prevention and detection

Question sources

Computer Science MCQ practice collection · Question 573

Question 46 | AM0574

Which listed protocol sends or relays email messages between mail systems?

A. FTP

B. HTTP

C. SMTP

D. DHCP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C: SMTP

SMTP is the mail sending and relay protocol. HTTP serves web resources, so a webmail interface can use HTTP without making it the mail-transfer protocol requested here. FTP transfers files as its named service, and DHCP supplies host configuration. Distinguishing the mail transport role from a user-facing web interface removes the broad interpretation of providing email services.

Why each choice is right or wrong

A - Incorrect: FTP is a file-transfer protocol, not the listed mail relay protocol.

B - Incorrect: HTTP serves web resources; webmail access is distinct from mail-system relay.

C - Correct: SMTP sends and relays email messages.

D - Incorrect: DHCP supplies host configuration rather than transferring email.

Primary note proof

Study-note section NSA10

> DNS resolves names; DHCP supplies host configuration. SMTP sends/relays mail, while IMAP/POP retrieve it. FTP is a named file-transfer protocol; HTTP also transfers files, so “transfers files” is not an exclusive FTP description. HTTP/HTTPS serve web resources. SSH provides encrypted remote login; SFTP transfers files over SSH and differs from FTP or FTPS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

> Application Layer
> The Application Layer is a combination of Session,
> Presentation & Application Layers of OSI models and defines
> high level protocols like File Transfer (FTP), Electronic Mail
> (SMTP), Virtual Terminal (TELNET), Domain Name Service
> (DNS), etc.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

> The Process/Application Layer Protocols
> • Telnet
> • File Transfer Protocol (FTP)
> • Network File System (NFS).
> • Simple Mail Transfer Protocol (SMTP
> • POP3 is used to receive mail.
> • Simple Network Management Protocol (SNMP)
> • Domain Name Service (DNS)
> • Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science MCQ practice collection · Question 574

Question 47 | AM0575

Which service translates domain names into IP addresses?

A. DNS

B. DHCP

C. FTP

D. HTTP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: DNS

Name-to-address translation is forward DNS resolution. NSA06 specifies A records for IPv4 and AAAA for IPv6. DHCP instead provides host configuration, while FTP and HTTP provide application data-transfer services. Those functions are distinct from resolving a domain name into its address records.

Why each choice is right or wrong

A - Correct: DNS forward resolution maps names to address records, using A or AAAA as appropriate.

B - Incorrect: DHCP supplies client addresses and options rather than translating arbitrary domain names.

C - Incorrect: FTP transfers files and does not provide the domain-name resolution service.

D - Incorrect: HTTP exchanges web resources rather than serving as the name-to-address directory service.

Primary note proof

Study-note section NSA06

> Forward resolution obtains an address from a name, using A for IPv4 and AAAA for IPv6. Reverse lookup uses a PTR record in the reverse namespace to obtain a name from an address. CNAME aliases a name, MX identifies mail exchangers and NS identifies authoritative name servers. Forward and reverse records need not both exist or be perfect inverses.

Study-note section NSA10

> DNS resolves names; DHCP supplies host configuration. SMTP sends/relays mail, while IMAP/POP retrieve it. FTP is a named file-transfer protocol; HTTP also transfers files, so “transfers files” is not an exclusive FTP description. HTTP/HTTPS serve web resources. SSH provides encrypted remote login; SFTP transfers files over SSH and differs from FTP or FTPS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

/etc/hosts
The /etc/hosts file always contains the localhost IP
address, 127.0.0.1, which is
usedfor interprocess communication.
Never remove this line! Sometimes contains
addresses of additional hosts, whichcan be contacted
without using an external naming service such as
DNS (the Domain Name Server).

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science MCQ practice collection · Question 575

Question 48 | AM0576

Which listed protocols transfer file content from a server to a client as part of their ordinary service? Select all that apply.

A. SMTP

B. DNS

C. FTP

D. HTTP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C, D: FTP; HTTP

FTP is specifically a file-transfer protocol, while HTTP also transfers file resources through web request and response exchanges. Their service semantics differ, but the broad content-transfer capability applies to both. SMTP is classified here as mail submission and relay, and DNS as name resolution. The revised selection instruction recognizes the two qualifying services without pretending that file transfer is exclusive to one protocol.

Why each choice is right or wrong

A - Incorrect: SMTP is identified with mail submission/relay rather than the file-transfer service in this comparison.

B - Incorrect: DNS resolves names and is not the offered file-content transfer service.

C - Correct: FTP is explicitly a named protocol for client/server file transfer.

D - Correct: HTTP also transfers file resources, so the general function is not exclusive to FTP.

Primary note proof

Study-note section NSA10

> DNS resolves names; DHCP supplies host configuration. SMTP sends/relays mail, while IMAP/POP retrieve it. FTP is a named file-transfer protocol; HTTP also transfers files, so “transfers files” is not an exclusive FTP description. HTTP/HTTPS serve web resources. SSH provides encrypted remote login; SFTP transfers files over SSH and differs from FTP or FTPS.

Study-note section NSA06

> Forward resolution obtains an address from a name, using A for IPv4 and AAAA for IPv6. Reverse lookup uses a PTR record in the reverse namespace to obtain a name from an address. CNAME aliases a name, MX identifies mail exchangers and NS identifies authoritative name servers. Forward and reverse records need not both exist or be perfect inverses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

> Application Layer
> The Application Layer is a combination of Session,
> Presentation & Application Layers of OSI models and defines
> high level protocols like File Transfer (FTP), Electronic Mail
> (SMTP), Virtual Terminal (TELNET), Domain Name Service
> (DNS), etc.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

> The Process/Application Layer Protocols
> • Telnet
> • File Transfer Protocol (FTP)
> • Network File System (NFS).
> • Simple Mail Transfer Protocol (SMTP

- POP3 is used to receive mail.
- Simple Network Management Protocol (SNMP)
- Domain Name Service (DNS)
- Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science MCQ practice collection · Question 576

Question 49 | AM0577

Which service is responsible for assigning IP addresses to devices on a network?

A. DNS

B. DHCP

C. FTP

D. HTTP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer B: DHCP

NSA04 assigns host-address configuration to DHCP, including a server-managed scope and time-limited leases. DNS resolves names, while FTP and HTTP transfer application content. Therefore DHCP is the network service responsible for automatically supplying a device's IP configuration among these options.

Why each choice is right or wrong

A - Incorrect: DNS resolves names into records; it does not lease the device's own network address.

B - Correct: DHCP supplies addresses and configuration options to clients from a managed scope.

C - Incorrect: FTP transfers files rather than assigning network addresses to devices.

D - Incorrect: HTTP provides web request/response services and not host-address allocation.

Primary note proof

Study-note section NSA04

DHCP supplies addresses and configuration options. Typical initial IPv4 allocation is Discover→Offer→Request→ACK. The server manages its scope/pool, allocated leases, exclusions and reservations; options may include mask, gateway and DNS servers. A dynamic lease lasts for a specified interval, not forever.

Study-note section NSA10

DNS resolves names; DHCP supplies host configuration. SMTP sends/relays mail, while IMAP/POP retrieve it. FTP is a named file-transfer protocol; HTTP also transfers files, so “transfers files” is not an exclusive FTP description. HTTP/HTTPS serve web resources. SSH provides encrypted remote login; SFTP transfers files over SSH and differs from FTP or FTPS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science MCQ practice collection · Question 577

Question 50 | AM0578

Which protocol is used for secure remote login and file transfer?

A. SSH

B. DNS

C. FTP

D. HTTP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: SSH, with SFTP for file transfer over SSH

NSA10 assigns protected remote login to SSH and file transfer over that protected connection to SFTP. Thus SSH is the underlying secure protocol family among these options. DNS provides naming; ordinary FTP and HTTP do not by themselves provide the protected remote-login capability in the stem.

Why each choice is right or wrong

A - Correct: SSH protects remote login and supports the SFTP file-transfer service over SSH.

B - Incorrect: DNS resolves names and supplies neither remote terminal login nor a secure file-transfer session.

C - Incorrect: Ordinary FTP transfers files but is distinct from SFTP and does not supply encrypted remote login.

D - Incorrect: Ordinary HTTP supplies web exchange semantics, not the SSH remote-login and SFTP functions.

Primary note proof

Study-note section NSA10

DNS resolves names; DHCP supplies host configuration. SMTP sends/relays mail, while IMAP/POP retrieve it. FTP is a named file-transfer protocol; HTTP also transfers files, so “transfers files” is not an exclusive FTP description. HTTP/HTTPS serve web resources. SSH provides encrypted remote login; SFTP transfers files over SSH and differs from FTP or FTPS.

Study-note section NSA30

Use authenticated protected remote administration such as SSH or properly secured management services, with restricted source access and auditing. VPNs provide protected connectivity under their defined security model; they are not packet analyzers. Firewall filtering permits/blocks traffic according to policy; it is not inherently encryption. Losing remote connectivity while changing a management route/firewall is an operational risk to plan for.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 58 | lines 22-32

The following are some of proprietary and open source
applications that can be used to achieve this.
• SSH (Secure Shell)
• OpenSSH (OpenBSD Secure Shell)
• Telnet
• rlogin
• rsh
• VNC (Virtual Network Computing)
• FreeNX
• Wireless Remote Administration
• Remote Desktop Connection (RDC)

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

Application Layer
The Application Layer is a combination of Session,
Presentation & Application Layers of OSI models and defines
high level protocols like File Transfer (FTP), Electronic Mail
(SMTP), Virtual Terminal (TELNET), Domain Name Service
(DNS), etc.

Question sources

Computer Science MCQ practice collection · Question 578

Question 51 | AM0579

Which service is used for translating private IP addresses to public IP addresses?

A. NAT

B. DNS

C. FTP

D. HTTP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: NAT

NAT rewrites network addresses, commonly translating private internal IPv4 addresses for access to a public network. Routing still selects the path and NAT does not replace firewall or authentication policy. DNS resolves names, FTP transfers files, and HTTP serves web resources. Those application services may run through a translating router, but they do not perform the address-rewriting function themselves.

Why each choice is right or wrong

A - Correct: NAT performs the requested address translation.

B - Incorrect: DNS maps names and related records rather than rewriting packet addresses.

C - Incorrect: FTP transfers files rather than translating network addresses.

D - Incorrect: HTTP serves web resources rather than rewriting IP addresses.

Primary note proof

Study-note section NSA11

> Routing selects a next hop using a destination prefix; NAT rewrites addresses. Many-to-one address-and-port translation (PAT/NAPT) maps internal flows to one public address using distinct port bindings. For example, two private client sockets can appear as public-address ports 40001 and 40002, with replies translated back to the correct clients.

Study-note section NSA11

> NAT is often used when private IPv4 clients access public networks; it does not itself replace a gateway, route, firewall or authentication. One-to-one NAT and destination translation are other forms, so NAT does not always mean many hosts share one address. Public addressing inside a LAN does not automatically require NAT. Unsolicited inbound service access may require explicit mapping and filtering.

Study-note section NSA06

Forward resolution obtains an address from a name, using A for IPv4 and AAAA for IPv6. Reverse lookup uses a PTR record in the reverse namespace to obtain a name from an address. CNAME aliases a name, MX identifies mail exchangers and NS identifies authoritative name servers. Forward and reverse records need not both exist or be perfect inverses.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Question sources

Computer Science MCQ practice collection · Question 579

Question 52 | AM0580

Which protocol is used for web browsing?

A. SMTP

B. DNS

C. FTP

D. HTTP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: HTTP

Web browsing obtains resources through web request/response exchanges. NSA10 identifies HTTP/HTTPS for these resources, making HTTP the offered protocol. SMTP handles mail submission/relay, DNS handles name resolution and FTP handles its own file-transfer service; they may assist applications but do not define the web exchange asked for.

Why each choice is right or wrong

A - Incorrect: SMTP submits and relays email rather than serving browser web resources.

B - Incorrect: DNS resolves server names and is a supporting service rather than the web resource-transfer protocol.

C - Incorrect: FTP is a separate file-transfer protocol, not the normal web-browsing exchange named in the note.

D - Correct: HTTP supplies web-resource request/response exchanges and is explicitly named for web services.

Primary note proof

Study-note section NSA10

> DNS resolves names; DHCP supplies host configuration. SMTP sends/relays mail, while IMAP/POP retrieve it. FTP is a named file-transfer protocol; HTTP also transfers files, so “transfers files” is not an exclusive FTP description. HTTP/HTTPS serve web resources. SSH provides encrypted remote login; SFTP transfers files over SSH and differs from FTP or FTPS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

> Application Layer
> The Application Layer is a combination of Session,
> Presentation & Application Layers of OSI models and defines
> high level protocols like File Transfer (FTP), Electronic Mail
> (SMTP), Virtual Terminal (TELNET), Domain Name Service
> (DNS), etc.

Data Communication and Computer Networking module | PDF page 28 | lines 2-7

> TCP and UDP must use port numbers to communicate
> with the upper layers, because they‘re what keep track of
> different conversations crossing the network
> simultaneously. These port numbers identify the source
> and destination application or process in the TCP
> segment. There are 216 = 65,536 ports available.

Question sources

Computer Science MCQ practice collection · Question 580

Question 53 | AM0581

Which service provides remote access to a network from a different location?

A. VPN

B. DNS

C. FTP

D. HTTP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: VPN

A remote user can extend private-network connectivity through a VPN over another network, matching the service in the stem. DNS performs lookup, FTP transfers files and HTTP serves web resources. Those can be reachable remotely but do not by themselves extend private network access as the VPN service does.

Why each choice is right or wrong

A - Correct: A VPN extends private connectivity across another network to support remote access.

B - Incorrect: DNS supplies name resolution rather than a remote private-network connection.

C - Incorrect: FTP supplies access to a file-transfer service rather than general remote network connectivity.

D - Incorrect: HTTP supplies web-resource access rather than the private-network extension described.

Primary note proof

Study-note section NSA10

> A VPN extends private connectivity over another network; secure tunneling commonly encrypts/authenticates traffic but depends on protocol/configuration. RTP carries real-time media timing/sequence information, often over UDP; it does not guarantee timely delivery. Network printers expose services shared by clients. Provide accounts, permissions, configuration, monitoring and recovery for each service, not just its protocol name.

Study-note section NSA30

> Use authenticated protected remote administration such as SSH or properly secured management services, with restricted source access and auditing. VPNs provide protected connectivity under their defined security model; they are not packet analyzers. Firewall filtering permits/blocks traffic according to policy; it is not inherently encryption. Losing remote connectivity while changing a management route/firewall is an operational risk to plan for.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 57 | lines 78-87

Remote Administration
Remote administration is an approach being followed to control either a computer system or a network or an application or all three from a remote location. Simply put, Remote administration refers to any method of controlling a computer from a remote location. A remote location may refer to a computer in the next room or one on the other side of the world. It may also refer to both legal and illegal remote administration. Generally, remote administration is essentially adopted when it is difficult or

Data Communication and Computer Networking module | PDF page 58 | lines 22-32

The following are some of proprietary and open source applications that can be used to achieve this.
• SSH (Secure Shell)
• OpenSSH (OpenBSD Secure Shell)
• Telnet
• rlogin
• rsh
• VNC (Virtual Network Computing)
• FreeNX
• Wireless Remote Administration
• Remote Desktop Connection (RDC)

Question sources

Computer Science MCQ practice collection · Question 581

Question 54 | AM0582

Which protocol is used for real-time communication, such as video conferencing?

A. SMTP

B. DNS

C. FTP

D. RTP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: RTP

NSA10 identifies RTP as carrying real-time media timing and sequence information, commonly over UDP. That is the relevant protocol for conference media among the options. The answer does not imply guaranteed deadlines: actual delivery depends on the network and application. SMTP, DNS and FTP have unrelated mail, naming and file-transfer roles.

Why each choice is right or wrong

A - Incorrect: SMTP handles mail submission/relay, not real-time conference media timing and sequencing.

B - Incorrect: DNS resolves names and is not the media transport protocol described.

C - Incorrect: FTP transfers files rather than carrying the real-time media timing/sequence service.

D - Correct: RTP carries real-time media timing and sequence information, including uses such as video conferencing.

Primary note proof

Study-note section NSA10

> A VPN extends private connectivity over another network; secure tunneling commonly encrypts/authenticates traffic but depends on protocol/configuration. RTP carries real-time media timing/sequence information, often over UDP; it does not guarantee timely delivery. Network printers expose services shared by clients. Provide accounts, permissions, configuration, monitoring and recovery for each service, not just its protocol name.

Study-note section NSA10

> DNS resolves names; DHCP supplies host configuration. SMTP sends/relays mail, while IMAP/POP retrieve it. FTP is a named file-transfer protocol; HTTP also transfers files, so “transfers files” is not an exclusive FTP description. HTTP/HTTPS serve web resources. SSH provides encrypted remote login; SFTP transfers files over SSH and differs from FTP or FTPS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 18 | lines 27-32

> Application Layer
> The Application Layer is a combination of Session,
> Presentation & Application Layers of OSI models and defines
> high level protocols like File Transfer (FTP), Electronic Mail
> (SMTP), Virtual Terminal (TELNET), Domain Name Service
> (DNS), etc.

Question sources

Computer Science MCQ practice collection · Question 582

Question 55 | AM0583

Which service, using its many-to-one address-and-port translation form, lets several private IPv4 devices share one public IPv4 address?

A. NAT

B. DNS

C. FTP

D. HTTP

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: NAT

Many-to-one NAT with port translation maintains separate bindings for internal flows. Multiple clients can therefore appear under one public address while distinct translated ports allow replies to return to the right client. This is a particular form of NAT, not a property of every NAT configuration. DNS, FTP and HTTP supply naming, file-transfer and web services rather than the address-sharing mechanism.

Why each choice is right or wrong

A - Correct: NAT in its PAT/NAPT form maps multiple internal flows to one public address.

B - Incorrect: DNS resolves names rather than maintaining address-and-port translation bindings.

C - Incorrect: FTP transfers files; it is not the address-sharing mechanism.

D - Incorrect: HTTP serves web resources; it does not provide the translation bindings.

Primary note proof

Study-note section NSA11

> Routing selects a next hop using a destination prefix; NAT rewrites addresses. Many-to-one address-and-port translation (PAT/NAPT) maps internal flows to one public address using distinct port bindings. For example, two private client sockets can appear as public-address ports 40001 and 40002, with replies translated back to the correct clients.

Study-note section NSA11

> NAT is often used when private IPv4 clients access public networks; it does not itself replace a gateway, route, firewall or authentication. One-to-one NAT and destination translation are other forms, so NAT does not always mean many hosts share one address. Public addressing inside a LAN does not automatically require NAT. Unsolicited inbound service access may require explicit mapping and filtering.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Question sources

Computer Science MCQ practice collection · Question 583

Question 56 | AM0584

Which policy conflicts with restricting access to a protected internal network to authorized users, assuming the wireless service directly admits users to that network?

A. Password complexity requirements

B. Firewall configuration

C. Regular data backups

D. Open wireless network access

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer D: Open wireless network access

A protected internal network requires controlled admission. Password requirements, appropriately configured firewall rules and protected regular backups can implement security and recovery policy. Unrestricted open wireless admission to the same protected network bypasses that admission requirement. An intentionally isolated guest service would be a different case; it is excluded by the stated direct-access condition.

Why each choice is right or wrong

A - Incorrect: Password requirements support authentication and controlled access.

B - Incorrect: Firewall policy can restrict access to authorized systems.

C - Incorrect: Regular backups support recovery and availability.

D - Correct: Unrestricted open wireless admission bypasses authorization for the protected network.

Primary note proof

Study-note section NSA26

A security policy states required behavior, scope, accountable roles and enforcement. Begin with assets/service objectives, assess threats and weaknesses, choose controls, define exceptions/incident procedures, approve/communicate, monitor and periodically review. CIA means confidentiality, integrity and availability; neither sensitive-data protection nor availability alone is the universal primary goal.

Study-note section NSA26

A policy can require password controls, approved encryption, restricted administrative access, backups and patch deadlines; firewall rules implement part of it. Open unrestricted wireless access conflicts with controlled internal access unless intentionally isolated as a guest service. “Use the system passively” is not a purpose of security policy. Policies coordinate action and reduce risk; they do not guarantee every attack will fail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 30 | lines 6-8

• Backing up the network software and data regularly to protect against the failure of network or any of its components

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Question sources

Computer Science MCQ practice collection · Question 584

Question 57 | AM0585

Which purpose is directly served when a firewall applies permit/deny rules to incoming network traffic?

A. To prevent unauthorized access to the network

B. To encrypt network traffic

C. To manage network bandwidth

D. To monitor network performance

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer A: To prevent unauthorized access to the network

A firewall applies policy to allow permitted traffic and reject disallowed access. This directly supports preventing unauthorized network access. Encryption protects confidentiality, bandwidth management allocates or shapes traffic capacity, and performance monitoring measures service behavior. A device may combine these features, but the stated permit/deny operation is access enforcement rather than those separate functions.

Why each choice is right or wrong

A - Correct: Permit/deny filtering directly prevents disallowed network access.

B - Incorrect: Encryption protects confidentiality and is not inherent in traffic filtering.

C - Incorrect: Bandwidth management controls traffic capacity rather than authorization.

D - Incorrect: Performance monitoring measures behavior rather than enforcing permit/deny policy.

Primary note proof

Study-note section NSA30

> Use authenticated protected remote administration such as SSH or properly secured management services, with restricted source access and auditing. VPNs provide protected connectivity under their defined security model; they are not packet analyzers. Firewall filtering permits/blocks traffic according to policy; it is not inherently encryption. Losing remote connectivity while changing a management route/firewall is an operational risk to plan for.

Study-note section NSA28

> Encryption provides confidentiality with correct key/nonce use; authenticated encryption adds integrity/authenticity. Unspecified encryption alone does not guarantee both properties. A password hash supports verification without recovering the original password; it is not a

reversible credential store. Encryption can protect a recoverable credential only if its key is separately protected.

Study-note section NSA12

QoS classifies, queues and prioritizes traffic to meet performance objectives such as low delay or limited loss. It can operate across layers and support applications, so it is not exclusively an application-layer service. Installing a new service is an administrative action; email configuration is another action and proxies/agents are components. Distinguish service-management actions from components using a stated taxonomy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Question sources

Computer Science MCQ practice collection · Question 585

Question 58 | AM0586

Which of the following is an example of a network security vulnerability?

A. Weak password policies

B. Regular software updates

C. Use of strong encryption algorithms

D. Implementing access control lists

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer A: Weak password policies

NSA27 identifies weak passwords as exploitable weaknesses. A policy permitting them creates the corresponding vulnerability. Regular updates, appropriate encryption and access-control lists are protective measures when correctly deployed, rather than examples of the weakness requested. They are not guarantees of security, but their stated role here is defensive.

Why each choice is right or wrong

A - Correct: Weak password policy permits easily compromised credentials and creates an exploitable access weakness.

B - Incorrect: Regular updates address known software weaknesses rather than constituting the vulnerability example requested.

C - Incorrect: Strong appropriately used encryption is a protective measure rather than an example of weak authentication policy.

D - Incorrect: Access-control lists restrict permitted actions/traffic and are a defensive control when properly configured.

Primary note proof

Study-note section NSA27

> Hardening reduces exposure through unnecessary-service removal, supported patches, restricted accounts/permissions and appropriate port filtering. Blocking an unused known port can be hardening; blocking every useful port merely destroys availability. Weak passwords and unpatched exploitable software are vulnerabilities; attacks exploit weaknesses, and risk combines likelihood with impact.

Study-note section NSA28

> Encryption provides confidentiality with correct key/nonce use; authenticated encryption adds integrity/authenticity. Unspecified encryption alone does not guarantee both properties. A password hash supports verification without recovering the original password; it is not a reversible credential store. Encryption can protect a recoverable credential only if its key is separately protected.

Study-note section NSA30

> Segmentation separates groups of systems and enforces allowed flows with routing/filtering policies; VLAN membership alone does not establish complete security. NAC assesses device/user admission and can reject or quarantine unauthorized/noncompliant devices. Authentication proves an identity, while access rules decide which resources it may use.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

> • Putting security procedures in place to make
> certain that only authorized users have access to
> the network and ensuring that all security

procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 42 | lines 21-28

Administrator: The —Administrator‖ user account has complete control over the PC. He or she can install anything and make changes that affect all users of that PC.
Standard: The —Standard‖ user account can only use the software that‘s already installed by the administrator and change system settings that don‘t affect other users.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

System Monitoring Software Essentials
In order to keep your system fit for purpose, your monitoring activities need to cover the following priorities:
• Acceptable delivery speeds
• Constant availability
• Preventative maintenance
• Software version monitoring and patching
• Intrusion detection
• Data integrity
• Security monitoring
• Attack mitigation
• Virus prevention and detection

Question sources

Computer Science MCQ practice collection · Question 586

Question 59 | AM0587

What is the purpose of intrusion detection systems (IDS)?

A. To prevent network attacks

B. To monitor network traffic for suspicious activities

C. To encrypt network communications

D. To block malicious software

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer B: Monitor for suspicious activity and alert

NSA23 distinguishes detection from prevention. An IDS monitors for suspicious activity and generates detection/alerts; blocking inline is an IPS capability. Encryption is a separate data-protection mechanism. Although IDS findings can support a broader prevention program, direct prevention or malware blocking is not the defining IDS role stated here.

Why each choice is right or wrong

A - Incorrect: Direct blocking/prevention is the IPS distinction; an IDS primarily detects and alerts.

B - Correct: Monitoring traffic for suspicious behavior matches the IDS detection/alert role.

C - Incorrect: Encrypting communications is a confidentiality mechanism, not intrusion detection.

D - Incorrect: Blocking malicious software is a preventive action, not the defining monitoring/alert purpose of an IDS.

Primary note proof

Study-note section NSA23

> An IDS detects/alerts; an IPS can block inline. Scanning identifies potential weaknesses rather than proving successful exploitation. A captured packet, open port or alert needs context. Monitoring network usage is not the same as configuring access control or encrypting communications.

Study-note section NSA28

> Encryption provides confidentiality with correct key/nonce use; authenticated encryption adds integrity/authenticity. Unspecified encryption alone does not guarantee both properties. A password hash supports verification without recovering the original password; it is not a reversible credential store. Encryption can protect a recoverable credential only if its key is separately protected.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

> System Monitoring Software Essentials
> In order to keep your system fit for purpose, your monitoring activities need to cover the following priorities:
> • Acceptable delivery speeds
> • Constant availability
> • Preventative maintenance
> • Software version monitoring and patching
> • Intrusion detection

- Data integrity
- Security monitoring
- Attack mitigation
- Virus prevention and detection

Data Communication and Computer Networking module | PDF page 54 | lines 13-17

Bandwidth monitoring tools and NetFlow and sFlow based traffic analyzers help you stay aware of the activity, capacity, and health of your network. They allow you to watch traffic as it flows through routers and switches, or arrive at and leaves hosts.

Question sources

Computer Science MCQ practice collection · Question 587

Question 60 | AM0588

During troubleshooting, which action tests IP reachability with ICMP echo messages without changing the system or network configuration?

A. Configuring a firewall rule

B. Conducting a security audit

C. Installing antivirus software

D. Ping testing network connectivity

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: Ping testing network connectivity

A ping test gathers reachability evidence by exchanging ICMP echo messages. It is a diagnostic observation rather than a configuration change. Firewall rule changes can be legitimate fixes for a diagnosed filtering fault, while audits and antivirus installation serve other security purposes. None of those alternatives performs the specific echo test requested. A failed test must still be interpreted alongside filtering and loss.

Why each choice is right or wrong

A - Incorrect: Changing a firewall rule changes policy rather than performing the specified observation.

B - Incorrect: A security audit assesses controls; it is not an ICMP echo test.

C - Incorrect: Installing antivirus changes host protection rather than sending echo probes.

D - Correct: Ping performs the specified reachability test without changing network policy.

Primary note proof

Study-note section NSA08

> Define the symptom, scope, time and recent changes; gather the configuration, baseline and logs. Form a testable hypothesis, test the least disruptive explanation, isolate the failing component, apply a targeted fix, verify the user service and document the result. Change one relevant factor at a time when possible. A reboot can hide evidence and is not a diagnosis by itself.

Study-note section NSA-S04

> Ping uses ICMP echo messages to test IP reachability. A reply shows that the addressed host and path support that exchange; no reply alone does not prove the host is disconnected, because filtering or loss can suppress responses. Traceroute instead discovers responding intermediate hops through packet-lifetime expiry; it is a path-discovery diagnostic, not the same question as an end-to-end echo test.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 27 | lines 26-28

> Ping (Packet Internet Groper) uses ICMP echo
> messages to check the physical and logical
> connectivity of machineson a network.

Data Communication and Computer Networking module | PDF page 30 | lines 16-18

> • Diagnosing and troubleshooting problems on the
> network and determining the best course of action
> to take to solve them.

Question sources

Computer Science MCQ practice collection · Question 588

Question 61 | AM0589

Under the note’s primary tool-role classification, which is the conventional general-purpose packet capture and dissection analyzer?

A. Wireshark

B. Nmap

C. Snort

D. Nessus

Hint

Match the requested observation to the tool’s actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer A: Wireshark

Wireshark is identified as the conventional analyzer for capturing and dissecting packets. Snort also has sniffing and logging capabilities, but its primary role in this comparison is suspicious-traffic detection and alerting. Nmap discovers hosts, ports and services, while Nessus assesses vulnerabilities and configuration. The primary-role scope distinguishes the intended analyzer without incorrectly denying overlapping packet capabilities in other tools.

Why each choice is right or wrong

A - Correct: Wireshark captures and dissects packets and is a conventional general packet analyzer.

B - Incorrect: Nmap performs host, port and service discovery rather than general packet capture/dissection.

C - Incorrect: Snort can sniff and log, but is classified primarily here by suspicious-traffic detection rather than as the conventional general analyzer.

D - Incorrect: Nessus is identified with vulnerability/configuration assessment rather than the packet-analyzer role.

Primary note proof

Study-note section NSA23

> Task Manager shows running processes/resource use; Resource Monitor/Performance Monitor give more detailed counters. Wireshark captures and dissects packets. Nmap discovers hosts/ports/services. Nessus assesses vulnerabilities/configuration. Snort can detect suspicious traffic and alert, and can prevent traffic in suitable inline deployments; it also has sniffing/logging roles. Thus packet-capture capability is not exclusive to one product, although Wireshark is the conventional analyzer choice.

Study-note section NSA23

> An IDS detects/alerts; an IPS can block inline. Scanning identifies potential weaknesses rather than proving successful exploitation. A captured packet, open port or alert needs context. Monitoring network usage is not the same as configuring access control or encrypting communications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 54 | lines 13-17

> Bandwidth monitoring tools and NetFlow and sFlow
> based traffic analyzers help you stay aware of the activity,
> capacity, and health of your network. They allow you to
> watch traffic as it flows through routers and switches, or
> arrive at and leaves hosts.

Question sources

Computer Science MCQ practice collection · Question 589

Question 62 | AM0590

Which of the following is an example of a denial-of-service (DoS) attack?

A. Man-in-the-middle attack

B. Phishing attack

C. Distributed denial-of-service (DDoS) attack

D. Brute-force attack

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer C: Distributed denial-of-service (DDoS)

DDoS is denial of service performed from distributed sources, so it is directly a type of DoS. Interception, phishing and brute force describe different attack mechanisms; each might have secondary effects, but those names alone do not identify the service-availability attack requested. NSA29 supplies the explicit DoS/DDoS relationship.

Why each choice is right or wrong

A - Incorrect: An interception attack reads or alters communications; its name does not identify denial of service.

B - Incorrect: Phishing deceives users rather than specifically exhausting or denying a service.

C - Correct: DDoS is a denial-of-service attack using distributed sources.

D - Incorrect: Brute force tries credentials; that mechanism is distinct from the denial-of-service category asked about.

Primary note proof

Study-note section NSA29

> DoS harms service availability; DDoS uses distributed sources. Phishing deceives users, brute force tries credentials and an interception attack reads/alters communications. Correlate logs using time and identity/context. Preserve relevant evidence before resetting systems. A suspicious event is not automatically confirmed compromise; record facts and uncertainty.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

> System Monitoring Software Essentials
> In order to keep your system fit for purpose, your
> monitoring activities need to cover the following
> priorities:
> • Acceptable delivery speeds

- Constant availability
- Preventative maintenance
- Software version monitoring and patching
- Intrusion detection
- Data integrity
- Security monitoring
- Attack mitigation
- Virus prevention and detection

Question sources

Computer Science MCQ practice collection · Question 590

Question 63 | AM0591

For a secure VPN tunnel already providing private connectivity, which option identifies the mechanism that keeps intercepted tunnel payloads confidential?

A. To encrypt network traffic

B. To block unauthorized access

C. To manage network bandwidth

D. To analyze network performance

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: To encrypt network traffic

A secure VPN commonly encrypts and authenticates its tunnel traffic. Encryption protects the content against disclosure to an observer who intercepts packets, assuming appropriate configuration and keys. Admission rules can restrict who connects, but that is a different question from protecting intercepted payloads. Bandwidth management and performance analysis also serve different purposes. The secure-tunnel scope avoids claiming that every unqualified VPN automatically encrypts.

Why each choice is right or wrong

A - Correct: Encryption hides tunnel payload contents from an interceptor without the required keys.

B - Incorrect: Blocking unauthorized admission is distinct from protecting the content of already intercepted packets.

C - Incorrect: Bandwidth management is QoS/traffic management, not the defining private-connectivity purpose of a VPN.

D - Incorrect: Performance analysis is monitoring; VPN connectivity does not make it a packet or performance analyzer.

Primary note proof

Study-note section NSA10

A VPN extends private connectivity over another network; secure tunneling commonly encrypts/authenticates traffic but depends on protocol/configuration. RTP carries real-time media timing/sequence information, often over UDP; it does not guarantee timely delivery. Network printers expose services shared by clients. Provide accounts, permissions, configuration, monitoring and recovery for each service, not just its protocol name.

Study-note section NSA28

Encryption provides confidentiality with correct key/nonce use; authenticated encryption adds integrity/authenticity. Unspecified encryption alone does not guarantee both properties. A password hash supports verification without recovering the original password; it is not a reversible credential store. Encryption can protect a recoverable credential only if its key is separately protected.

Study-note section NSA12

QoS classifies, queues and prioritizes traffic to meet performance objectives such as low delay or limited loss. It can operate across layers and support applications, so it is not exclusively an application-layer service. Installing a new service is an administrative action; email configuration is another action and proxies/agents are components. Distinguish service-management actions from components using a stated taxonomy.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 57 | lines 78-87

Remote Administration
Remote administration is an approach being followed to control either a computer system or a network or an application or all three from a remote location. Simply put, Remote administration refers to any method of controlling a computer from a remote location. A remote location may refer to a computer in the next room or one on the other side of the world. It may also refer to both legal and illegal remote administration. Generally, remote administration is essentially adopted when it is difficult or

Data Communication and Computer Networking module | PDF page 58 | lines 22-32

The following are some of proprietary and open source applications that can be used to achieve this.
• SSH (Secure Shell)
• OpenSSH (OpenBSD Secure Shell)
• Telnet
• rlogin
• rsh
• VNC (Virtual Network Computing)

- FreeNX
- Wireless Remote Administration
- Remote Desktop Connection (RDC)

Question sources

Computer Science MCQ practice collection · Question 591

Question 64 | AM0592

Which of the following is an example of a security incident response action?

A. Regular software updates

B. Password policy enforcement

C. Incident investigation and analysis

D. Network configuration backup

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer C: Incident investigation and analysis

PDF p.30 requires personnel to respond quickly and effectively to security failures and to diagnose and troubleshoot problems. Investigation and analysis is the option that reacts to an incident rather than routine prevention or preservation.

Why each choice is right or wrong

A - Incorrect: Regular updating is a routine patching activity (p.30), rather than investigating an incident.

B - Incorrect: Password-policy enforcement is an ongoing access-security measure, not the incident investigation requested.

C - Correct: Investigation/analysis implements diagnosis and effective response after security failure (p.30).

D - Incorrect: Configuration backup preserves data and is not the investigative response action.

Primary note proof

Study-note section NSA29

Incident response prepares, detects/analyzes, contains, removes the cause, restores service and learns from the event. Investigation is reactive response; routine patching and backup are normally preventive/preparatory operations. Document timestamps, affected systems, evidence, impact, actions, responsible people and recovery results. An incident report may record policy violations, but its scope is broader than that alone.

Study-note section NSA27

Preventive controls restrict harmful actions, detective controls expose them and recovery controls restore service. Account lifecycle, backup procedures, device configuration and traffic analysis can all be security controls under a stated objective. A vulnerability scan checks known weakness patterns, not bandwidth. Test/stage patches, prepare rollback, deploy and verify; testing and installation are both legitimate administrator duties.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 30 | lines 16-18

• Diagnosing and troubleshooting problems on the network and determining the best course of action to take to solve them.

Question sources

Computer Science MCQ practice collection · Question 592

Question 65 | AM0593

Which tool is commonly used for network scanning and host discovery?

A. Wireshark

B. Nmap

C. Snort

D. Nessus

Hint

Match the requested observation to the tool’s actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer B: Nmap

The requested scanning and discovery role directly matches Nmap in NSA23. Wireshark captures/dissects packets, Snort detects suspicious traffic and Nessus assesses vulnerabilities/configurations. These capabilities can complement discovery, but Nmap is the tool explicitly named for the requested host/port/service scan rather than those other primary purposes.

Why each choice is right or wrong

A - Incorrect: Wireshark's identified role is packet capture/dissection rather than active host and port discovery.

B - Correct: Nmap is explicitly described as discovering hosts, ports and services.

C - Incorrect: Snort's identified role is traffic inspection/detection, not the host-discovery scanning tool requested.

D - Incorrect: Nessus assesses vulnerabilities/configuration; Nmap is the note's explicit general host/port discovery tool.

Primary note proof

Study-note section NSA23

> Task Manager shows running processes/resource use; Resource Monitor/Performance Monitor give more detailed counters. Wireshark captures and dissects packets. Nmap discovers hosts/ports/services. Nessus assesses vulnerabilities/configuration. Snort can detect suspicious traffic and alert, and can prevent traffic in suitable inline deployments; it also has sniffing/logging roles. Thus packet-capture capability is not exclusive to one product, although Wireshark is the conventional analyzer choice.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 54 | lines 13-17

> Bandwidth monitoring tools and NetFlow and sFlow
> based traffic analyzers help you stay aware of the activity,
> capacity, and health of your network. They allow you to
> watch traffic as it flows through routers and switches, or
> arrive at and leaves hosts.

Data Communication and Computer Networking module | PDF page 54 | lines 49-63

> • Monitoring a range of server metrics: availability,
> CPU usage, memory usage, disk space, response
> time, and upload/download rates. Monitoring
> CPU temperature and power supply voltages.
> • Monitoring applications. Using deep knowledge
> of common applications and services to monitor
> key server processes, including web servers,
> database servers, and application stacks.

• Automatically alerting you of problems, such as servers or network devices that are overloaded or down, or worrisome trends. Customized alerts that can use multiple methods to contact you - email, SMS text messages, pager, etc.
• Triggering actions in response to alerts, to handle certain classes of problems automatically.

Question sources

Computer Science MCQ practice collection · Question 593

Question 66 | AM0594

A policy is being developed specifically to prevent disclosure of confidential records to unauthorized people. Which listed goal directly matches that objective?

A. Ensuring system availability

B. Protecting sensitive data

C. Increasing network speed

D. Enhancing user experience

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer B: Protecting sensitive data

Security policies can support confidentiality, integrity and availability, with priorities set by organizational objectives. This scenario explicitly prioritizes preventing unauthorized disclosure of confidential records, which is sensitive-data protection. Availability remains a legitimate security goal but addresses continued service rather than disclosure. Network speed and user experience have different primary objectives. The explicit policy objective determines the selection without imposing one universal priority on all policies.

Why each choice is right or wrong

A - Incorrect: Availability concerns continued service rather than the specified disclosure objective.

B - Correct: Protecting sensitive data directly addresses the stated confidentiality goal.

C - Incorrect: Network speed is a performance objective, not the security-policy priority defined in the note.

D - Incorrect: User experience can affect usability but is not the stated universal security-policy goal.

Primary note proof

Study-note section NSA26

A security policy states required behavior, scope, accountable roles and enforcement. Begin with assets/service objectives, assess threats and weaknesses, choose controls, define exceptions/incident procedures, approve/communicate, monitor and periodically review. CIA means confidentiality, integrity and availability; neither sensitive-data protection nor availability alone is the universal primary goal.

Study-note section NSA28

Encryption provides confidentiality with correct key/nonce use; authenticated encryption adds integrity/authenticity. Unspecified encryption alone does not guarantee both properties. A password hash supports verification without recovering the original password; it is not a reversible credential store. Encryption can protect a recoverable credential only if its key is separately protected.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

System Monitoring Software Essentials
In order to keep your system fit for purpose, your monitoring activities need to cover the following priorities:
• Acceptable delivery speeds
• Constant availability
• Preventative maintenance
• Software version monitoring and patching
• Intrusion detection
• Data integrity
• Security monitoring
• Attack mitigation
• Virus prevention and detection

Data Communication and Computer Networking module | PDF page 31 | lines 30-35

Principle 2 (Predictability)
The highest level aim in system administration is to work towards a

predictable system. Predictability has limits. It is the basis of reliability, hence trust and therefore security.

Question sources

Computer Science MCQ practice collection · Question 594

Question 67 | AM0595

Which of the following is an example of a security policy?

A. Setting strong password requirements

B. Monitoring network bandwidth

C. Updating software patches

D. Configuring firewall rules

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer A: Strong password requirements

A states a required behavior: passwords must satisfy specified strength requirements. That matches policy rather than an observation or operational change. Patching and firewall configuration implement controls and can be required by policy, but the options describe those actions without stating the governing requirement. Monitoring bandwidth is measurement rather than a policy statement.

Why each choice is right or wrong

A - Correct: Requiring strong passwords states mandatory behavior and is an explicit security-policy example.

B - Incorrect: Monitoring bandwidth is an operational measurement, not a stated security requirement.

C - Incorrect: Applying patches implements protection; a policy would state the required deadline/scope rather than merely the action.

D - Incorrect: Configuring rules implements policy, whereas the option does not itself state required behavior or governance.

Primary note proof

Study-note section NSA26

A security policy states required behavior, scope, accountable roles and enforcement. Begin with assets/service objectives, assess threats and weaknesses, choose controls, define exceptions/incident procedures, approve/communicate, monitor and periodically review. CIA

means confidentiality, integrity and availability; neither sensitive-data protection nor availability alone is the universal primary goal.

Study-note section NSA26

A policy can require password controls, approved encryption, restricted administrative access, backups and patch deadlines; firewall rules implement part of it. Open unrestricted wireless access conflicts with controlled internal access unless intentionally isolated as a guest service. “Use the system passively” is not a purpose of security policy. Policies coordinate action and reduce risk; they do not guarantee every attack will fail.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make
certain that only authorized users have access to
the network and ensuring that all security
procedures are followed
• Making sure the network personnel can respond
quickly and effectively in the event of a network
operational or security failure.

Data Communication and Computer Networking module | PDF page 31 | lines 17-29

Principle 1 (Policy is the foundation)
System administration begins with a
policy - adecision about what we want
and what should be, in relation to what we
can afford.
Policy speaks of what we wish to
accomplish with the system, and what we
are willing to tolerate of behavior within it.
It must refer to both the component parts and
to the environmentwith which the system
interacts. If we cannot secure predictability,
then we cannot expect
long-term conformance with a policy.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local
network to a single IP address. This address is often used
by the router that connects the computers to the Internet.
The router can be connected to a DSL modem, cable
modem, T1 line, or even a dial-up modem. When other
computers on the Internet attempt to access computers
within the local network, they only see the IP address of
the router. This adds an extra level of security, since the

router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Question sources

Computer Science MCQ practice collection · Question 595

Question 68 | AM0596

Which option states the most general purpose of access control, covering user permissions on logical resources as well as physical restrictions?

A. Limiting physical access to servers

B. Controlling user access to resources

C. Monitoring network traffic

D. Performing system backups

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer B: Controlling user access to resources

Access control decides which users or other subjects may use protected resources and which actions they may perform. Limiting physical entry to a server is one particular control, but does not express the broader account-and-resource permission function. Traffic monitoring observes activity, and backups preserve recoverable data. The general resource-access description therefore best covers the stated scope.

Why each choice is right or wrong

A - Incorrect: Physical server access restriction is a valid but narrower example.

B - Correct: Controlling user access to resources expresses the general purpose.

C - Incorrect: Monitoring observes activity rather than deciding resource permissions.

D - Incorrect: Backups preserve copies for recovery rather than granting access rights.

Primary note proof

Study-note section NSA17

Create named accounts, assign only needed groups/rights, disable departed users and audit privileged activity. Authentication verifies the account; authorization permits operations. A user right such as remote logon differs from a permission on one file. Group-based administration is easier to review than many hidden individual grants.

Study-note section NSA18

Windows file/folder permissions include read, write, read/execute, modify and full control, plus more granular rights such as delete. Inherited and explicit entries, group membership, deny entries and object scope affect effective access; use the effective-access view for a real account rather than adding labels informally. Network access can be constrained by both share and filesystem permissions.

Study-note section NSA30

Segmentation separates groups of systems and enforces allowed flows with routing/filtering policies; VLAN membership alone does not establish complete security. NAC assesses device/user admission and can reject or quarantine unauthorized/noncompliant devices. Authentication proves an identity, while access rules decide which resources it may use.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 43 | lines 2-10

What is File & Folder Permissions?
Permissions are a method for assigning access rights to specific user accounts and user groups. Through the use of permissions, Windows defines which user accounts and user groups can access which files and folders, and what they can do with them. To put it simply, permissions are the operating system‘s way of telling you what you can or cannot do with a file or folder.

Data Communication and Computer Networking module | PDF page 42 | lines 36-49

All user accounts have specific capabilities, privileges, and rights. When you create a user account, you can grant the user specific capabilities by making the user a member of one or more groups. This gives the user the capabilities of these groups. You then assign additional capabilities by making a user a member of the appropriate groups or withdraw capabilities by removing a user from a group. An important part of an administrator‘s job is being able to determine and set permissions, privileges, and logon rights as necessary. Although you can‘t change a group‘s built-in capabilities, you can change a group‘s default privileges and logon rights. For example, you could revoke network access to a computer by removing agroup‘s right to access the computer from the network.

Question sources

Computer Science MCQ practice collection · Question 596

Question 69 | AM0597

Which option describes a firewall's direct traffic-enforcement role, rather than observation, auditing or confidentiality protection?

A. Monitoring network traffic for security threats

B. Encrypting network communications

C. Blocking unauthorized access to the network

D. Auditing network activity

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer C: Blocking unauthorized access to the network

A firewall enforces traffic policy by permitting authorized flows and blocking disallowed ones. Monitoring and audit records can accompany this enforcement, but observing or recording activity does not itself express the requested permit-or-block action. Encryption protects communication confidentiality and is a separate function, even when a product combines it with firewall features.

Why each choice is right or wrong

A - Incorrect: Monitoring may accompany filtering, but observation is not the requested enforcement action.

B - Incorrect: Encryption protects confidentiality rather than deciding whether a flow is permitted.

C - Correct: Blocking unauthorized access directly enforces the traffic policy.

D - Incorrect: Auditing records or reviews activity rather than directly enforcing a traffic decision.

Primary note proof

Study-note section NSA30

> Use authenticated protected remote administration such as SSH or properly secured management services, with restricted source access and auditing. VPNs provide protected connectivity under their defined security model; they are not packet analyzers. Firewall filtering permits/blocks traffic according to policy; it is not inherently encryption. Losing remote connectivity while changing a management route/firewall is an operational risk to plan for.

Study-note section NSA23

> An IDS detects/alerts; an IPS can block inline. Scanning identifies potential weaknesses rather than proving successful exploitation. A captured packet, open port or alert needs context. Monitoring network usage is not the same as configuring access control or encrypting communications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

> Network Address Translation (NAT)
> NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Question sources

Computer Science MCQ practice collection · Question 597

Question 70 | AM0598

What is the purpose of vulnerability scanning in network and system administration?

A. Identifying security weaknesses and vulnerabilities

B. Encrypting sensitive data during transmission

C. Monitoring network bandwidth usage

D. Managing system logs

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer A: Identify potential security weaknesses

Vulnerability scanning assesses potential weaknesses and configuration problems, as NSA23 explains. It does not itself prove successful exploitation. Encryption, bandwidth measurement and log management are separate administration functions. Therefore A identifies the scan's purpose while retaining the note's caution that scan findings require contextual verification.

Why each choice is right or wrong

A - Correct: Finding potential security weaknesses and configuration vulnerabilities is the purpose assigned to vulnerability assessment.

B - Incorrect: Encryption protects data confidentiality and is not the act of checking for weaknesses.

C - Incorrect: Bandwidth monitoring measures traffic use rather than testing known vulnerability patterns.

D - Incorrect: Log management stores/organizes records and is distinct from scanning systems for weaknesses.

Primary note proof

Study-note section NSA23

Task Manager shows running processes/resource use; Resource Monitor/Performance Monitor give more detailed counters. Wireshark captures and dissects packets. Nmap discovers hosts/ports/services. Nessus assesses vulnerabilities/configuration. Snort can detect suspicious traffic and alert, and can prevent traffic in suitable inline deployments; it also has sniffing/logging roles. Thus packet-capture capability is not exclusive to one product, although Wireshark is the conventional analyzer choice.

Study-note section NSA23

An IDS detects/alerts; an IPS can block inline. Scanning identifies potential weaknesses rather than proving successful exploitation. A captured packet, open port or alert needs context. Monitoring network usage is not the same as configuring access control or encrypting communications.

Study-note section NSA27

Preventive controls restrict harmful actions, detective controls expose them and recovery controls restore service. Account lifecycle, backup procedures, device configuration and traffic analysis can all be security controls under a stated objective. A vulnerability scan checks known weakness patterns, not bandwidth. Test/stage patches, prepare rollback, deploy and verify; testing and installation are both legitimate administrator duties.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

System Monitoring Software Essentials
In order to keep your system fit for purpose, your monitoring activities need to cover the following priorities:
• Acceptable delivery speeds
• Constant availability
• Preventative maintenance
• Software version monitoring and patching
• Intrusion detection
• Data integrity
• Security monitoring
• Attack mitigation
• Virus prevention and detection

Data Communication and Computer Networking module | PDF page 53 | lines 29-43

In networking, it is typical to think of a system as a layered stack. User software sits at the top of the stack and system applications and services on the next layer down. Beneath the services and applications, you will encounter operating systems and firmware. Theperformance of software elements needs to be monitoredas an application stack. Users will notice performance problems with the software that they use, but those problems rarely arise within that software. All layers of the application stack need to be examined to find the root cause of performance issues. You need to head off problems with real-time statusmonitoring before they occur. Monitoring tools help you spot errors and service failures before they start to impactusers.

Question sources

Computer Science MCQ practice collection · Question 598

Question 71 | AM0599

Which of the following is a common vulnerability in computer systems?

A. Strong password policies

B. Regular software updates

C. Unpatched software vulnerabilities

D. Secure network segmentation

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer C: Unpatched exploitable software

NSA27 explicitly gives unpatched exploitable software as a vulnerability. Strong password policy, software updates and secure segmentation are defensive practices intended to reduce exposure. They can be imperfectly implemented, but as stated they are not the example of an exploitable weakness requested by the question.

Why each choice is right or wrong

A - Incorrect: Strong password policy is intended to reduce credential weaknesses rather than exemplify one.

B - Incorrect: Regular updates address known flaws and are a hardening practice.

C - Correct: Unpatched exploitable software contains a known weakness that an attack can exploit.

D - Incorrect: Secure segmentation restricts exposure and permitted flows rather than constituting the vulnerability example.

Primary note proof

Study-note section NSA27

> Hardening reduces exposure through unnecessary-service removal, supported patches, restricted accounts/permissions and appropriate port filtering. Blocking an unused known port can be hardening; blocking every useful port merely destroys availability. Weak passwords and unpatched exploitable software are vulnerabilities; attacks exploit weaknesses, and risk combines likelihood with impact.

Study-note section NSA30

> Segmentation separates groups of systems and enforces allowed flows with routing/filtering policies; VLAN membership alone does not establish complete security. NAC assesses device/user admission and can reject or quarantine unauthorized/noncompliant devices. Authentication proves an identity, while access rules decide which resources it may use.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 19-33

> Although the specifics of being a system administrator
> may change from platform to platform, there are
> underlying themes that do not. These themes make up the
> philosophy of system administration. The themes are:
> • Automate everything
> • Document everything
> • Communicate as much as possible
> • Know your resources
> • Know your users
> • Know your business
> • Security cannot be an afterthought
> • Plan ahead
> • Expect the unexpected
> • Backup and disaster recovery planning
> • Patching

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

> System Monitoring Software Essentials
> In order to keep your system fit for purpose, your
> monitoring activities need to cover the following
> priorities:
> • Acceptable delivery speeds
> • Constant availability
> • Preventative maintenance
> • Software version monitoring and patching

- Intrusion detection
- Data integrity
- Security monitoring
- Attack mitigation
- Virus prevention and detection

Question sources

Computer Science MCQ practice collection · Question 599

Question 72 | AM0600

What is the purpose of security incident response in network and system administration?

A. Preventing security incidents from occurring

B. Recovering data after a security breach

C. Analyzing network performance

D. Detecting and responding to security incidents

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer D: Detecting and responding to security incidents

PDF p.30 requires responding quickly and effectively to security failure, while p.54 includes intrusion detection and security monitoring. Detection and response therefore cover the requested process more fully than prevention alone or post-breach data recovery alone.

Why each choice is right or wrong

A - Incorrect: Prevention alone omits the explicit response to failures that occur (p.30).

B - Incorrect: Recovery is a narrower possible consequence; the response duty is broader (p.30).

C - Incorrect: Performance analysis addresses operational resources rather than the security-failure response.

D - Correct: Security detection/monitoring (p.54) plus responding to failures (p.30) jointly supports this.

Primary note proof

Study-note section NSA29

Incident response prepares, detects/analyzes, contains, removes the cause, restores service and learns from the event. Investigation is reactive response; routine patching and backup are normally preventive/preparatory operations. Document timestamps, affected systems, evidence, impact, actions, responsible people and recovery results. An incident report may record policy violations, but its scope is broader than that alone.

Study-note section NSA29

DoS harms service availability; DDoS uses distributed sources. Phishing deceives users, brute force tries credentials and an interception attack reads/alters communications. Correlate logs using time and identity/context. Preserve relevant evidence before resetting systems. A suspicious event is not automatically confirmed compromise; record facts and uncertainty.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

System Monitoring Software Essentials
In order to keep your system fit for purpose, your monitoring activities need to cover the following priorities:
• Acceptable delivery speeds
• Constant availability
• Preventative maintenance
• Software version monitoring and patching
• Intrusion detection
• Data integrity
• Security monitoring
• Attack mitigation
• Virus prevention and detection

Question sources

Computer Science MCQ practice collection · Question 600

Question 73 | AM0601

What protection does correctly implemented authenticated encryption provide for network data?

A. Preventing unauthorized access to the network

B. Monitoring network traffic for anomalies

C. Protecting data confidentiality and integrity

D. Performing system backups

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer C: Protecting data confidentiality and integrity

Encryption supplies confidentiality when keys and nonces are used correctly. Authenticated encryption also detects unauthorized alteration through integrity and authenticity protection. This explicit construction therefore supports both properties named together in the selected option. Network admission, traffic-anomaly monitoring and backup recovery are separate controls; none is performed merely by encrypting and authenticating a data payload.

Why each choice is right or wrong

A - Incorrect: Network admission/access control is distinct from encrypting content for confidentiality.

B - Incorrect: Anomaly detection is a monitoring function and does not describe the role of encryption.

C - Correct: Authenticated encryption combines confidentiality with integrity/authenticity protection.

D - Incorrect: Backups recover previous data and are not performed by encryption itself.

Primary note proof

Study-note section NSA28

> Encryption provides confidentiality with correct key/nonce use; authenticated encryption adds integrity/authenticity. Unspecified encryption alone does not guarantee both properties. A password hash supports verification without recovering the original password; it is not a reversible credential store. Encryption can protect a recoverable credential only if its key is separately protected.

Study-note section NSA30

> Segmentation separates groups of systems and enforces allowed flows with routing/filtering policies; VLAN membership alone does not establish complete security. NAC assesses device/user admission and can reject or quarantine unauthorized/noncompliant devices. Authentication proves an identity, while access rules decide which resources it may use.

Study-note section NSA24

> Backups recover previous data after deletion, compromise, corruption, failed upgrades/migrations or hardware loss. Authorization controls current access, a different function. A mirror/replica also copies deletions or corruption and does not replace versioned independent backups. Keep protected copies in separate failure domains, define retention and test restores.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

System Monitoring Software Essentials
In order to keep your system fit for purpose, your monitoring activities need to cover the following priorities:
• Acceptable delivery speeds
• Constant availability
• Preventative maintenance
• Software version monitoring and patching
• Intrusion detection
• Data integrity
• Security monitoring
• Attack mitigation
• Virus prevention and detection

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Question sources

Computer Science MCQ practice collection · Question 601

Question 74 | AM0602

Which listed activities can be security controls when implemented for the stated purpose: A recovery, B least-privilege account provisioning, C secure device configuration, D suspicious-traffic detection? Select all that apply.

A. Data backup and recovery procedures

B. User account creation

C. Network device configuration

D. Network traffic analysis

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer A, B, C, D: Data backup and recovery procedures; User account creation; Network device configuration; Network traffic analysis

Security controls include prevention, detection and recovery rather than only one technical mechanism. Backups support recovery, controlled account creation restricts access, secure device configuration reduces exposure, and traffic analysis can detect suspicious behavior. Each listed activity therefore qualifies under its explicit purpose. Simply performing an activity without an objective would not prove effectiveness, but the question supplies the security objective needed to classify it.

Why each choice is right or wrong

A - Correct: Backups and recovery procedures restore service/data after harmful events.

B - Correct: Controlled account creation implements identity lifecycle and access restriction.

C - Correct: Secure device configuration can reduce exposure and enforce permitted network behavior.

D - Correct: Traffic analysis is a detective control when used to identify suspicious behavior.

Primary note proof

Study-note section NSA27

> Preventive controls restrict harmful actions, detective controls expose them and recovery controls restore service. Account lifecycle, backup procedures, device configuration and traffic analysis can all be security controls under a stated objective. A vulnerability scan checks known weakness patterns, not bandwidth. Test/stage patches, prepare rollback, deploy and verify; testing and installation are both legitimate administrator duties.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 6-8

> • Backing up the network software and data
> regularly to protect against the failure of network
> or any of its components

Data Communication and Computer Networking module | PDF page 42 | lines 36-49

> All user accounts have specific capabilities, privileges,
> and rights. When you create a user account, you can grant
> the user specific capabilities by making the user a member
> of one or more groups. This gives the user the capabilities
> of these groups. You then assign additional capabilities by
> making a user a member of the appropriate groups or
> withdraw capabilities by removing a user from a group.
> An important part of an administrator‘s job is being able
> to determine and set permissions, privileges, and logon
> rights as necessary. Although you can‘t change a group‘s
> built-in capabilities, you can change a group‘s default

privileges and logon rights. For example, you could revoke network access to a computer by removing agroup‘s right to access the computer from the network.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Data Communication and Computer Networking module | PDF page 54 | lines 13-17

Bandwidth monitoring tools and NetFlow and sFlow based traffic analyzers help you stay aware of the activity, capacity, and health of your network. They allow you to watch traffic as it flows through routers and switches, or arrive at and leaves hosts.

Question sources

Computer Science MCQ practice collection · Question 602

Question 75 | AM0603

What is the purpose of a security incident report in network and system administration?

A. Documenting security policy violations

B. Sharing threat intelligence with other organizations

C. Investigating network performance issues

D. Identifying system vulnerabilities

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer A: Document security incidents, including policy violations

NSA29 assigns reports the documentation of affected systems, evidence, impact, actions and recovery results, and explicitly includes policy violations within that scope. A therefore states a legitimate report purpose among the choices, although it is not exhaustive. Threat sharing, performance investigation and weakness identification are separate activities that incident findings may later support.

Why each choice is right or wrong

A - Correct: Documenting policy violations is a valid part of incident reporting, although reports also cover impact and response.

B - Incorrect: External threat-intelligence sharing is a separate activity, not the documentation purpose established for this report.

C - Incorrect: Performance investigation is broader troubleshooting and does not specifically define a security incident report.

D - Incorrect: Finding vulnerabilities is assessment; incident reporting documents observed events, evidence and response.

Primary note proof

Study-note section NSA29

> Incident response prepares, detects/analyzes, contains, removes the cause, restores service and learns from the event. Investigation is reactive response; routine patching and backup are normally preventive/preparatory operations. Document timestamps, affected systems, evidence, impact, actions, responsible people and recovery results. An incident report may record policy violations, but its scope is broader than that alone.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

> • Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
> • Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 54 | lines 49-63

> • Monitoring a range of server metrics: availability, CPU usage, memory usage, disk space, response time, and upload/download rates. Monitoring CPU temperature and power supply voltages.
> • Monitoring applications. Using deep knowledge of common applications and services to monitor

key server processes, including web servers, database servers, and application stacks.
• Automatically alerting you of problems, such as servers or network devices that are overloaded or down, or worrisome trends. Customized alerts that can use multiple methods to contact you - email, SMS text messages, pager, etc.
• Triggering actions in response to alerts, to handle certain classes of problems automatically.

Question sources

Computer Science MCQ practice collection · Question 603

Question 76 | AM0604

Which actions can be valid troubleshooting steps when supported by a tested fault hypothesis and an appropriate recovery plan? Select all that apply.

A. Rebooting the system

B. Performing a software update

C. Resetting network configurations

D. Conducting a network scan

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A, B, C, D: Rebooting the system; Performing a software update; Resetting network configurations; Conducting a network scan

Rebooting, updating software, resetting configuration and scanning can each help with an appropriate diagnosed problem. Their value depends on the hypothesis and operational context: an update may address a known bug, a reset may correct damaged settings, and a scan may gather service evidence. None should be performed blindly. The question asks which actions can belong to troubleshooting, not which one should always be tried first.

Why each choice is right or wrong

A - Correct: Rebooting can be a justified troubleshooting action, although it can hide evidence and is not itself diagnosis.

B - Correct: An update can address a diagnosed software defect affecting network behavior.

C - Correct: Resetting incorrect network settings can resolve a configuration fault after preserving needed evidence.

D - Correct: A targeted network scan can gather reachability/service evidence for diagnosis.

Primary note proof

Study-note section NSA08

Define the symptom, scope, time and recent changes; gather the configuration, baseline and logs. Form a testable hypothesis, test the least disruptive explanation, isolate the failing component, apply a targeted fix, verify the user service and document the result. Change one relevant factor at a time when possible. A reboot can hide evidence and is not a diagnosis by itself.

Study-note section NSA08

Use bottom-up checks from physical link, top-down checks from application errors, or divide-and-conquer by testing a middle layer. Divide-and-conquer narrows the working/failing boundary; it is not random trial-and-error. Rebooting, updates, resetting settings and scanning can all be troubleshooting actions in suitable circumstances; use fault evidence to choose among them.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 16-18

• Diagnosing and troubleshooting problems on the network and determining the best course of action to take to solve them.

Data Communication and Computer Networking module | PDF page 53 | lines 29-43

In networking, it is typical to think of a system as a layered stack. User software sits at the top of the stack and system applications and services on the next layer down. Beneath the services and applications, you will encounter operating systems and firmware. Theperformance of software elements needs to be monitoredas an application stack. Users will notice performance problems with the software that they use, but those problems rarely arise within that software. All layers of the application stack need to be examined to find the root cause of performance issues. You need to head off problems with real-time statusmonitoring before they occur. Monitoring tools help you spot errors and service failures before they start to impactusers.

Data Communication and Computer Networking module | PDF page 30 | lines 19-33

Although the specifics of being a system administrator may change from platform to platform, there are underlying themes that do not. These themes make up the philosophy of system administration. The themes are:
• Automate everything
• Document everything

- Communicate as much as possible
- Know your resources
- Know your users
- Know your business
- Security cannot be an afterthought
- Plan ahead
- Expect the unexpected
- Backup and disaster recovery planning
- Patching

Question sources

Computer Science MCQ practice collection · Question 604

Question 77 | AM0605

Which purposes can analysis of existing system and network logs serve? Select all that apply.

A. Identifying potential security breaches

B. Monitoring network performance

C. Configuring firewall rules

D. Encrypting network communications

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A, B: Identifying potential security breaches; Monitoring network performance

Logs can reveal suspicious activity that suggests a security breach and can also expose timing, error or workload patterns related to performance. These are complementary uses of recorded observations. Changing firewall rules may follow the analysis, but it is a separate configuration action. Encryption protects communications in transit or storage rather than emerging from analysis of past records. Both analytical purposes therefore qualify.

Why each choice is right or wrong

A - Correct: Security events in logs can reveal potential breaches and support incident investigation.

B - Correct: Log timing/errors can reveal performance faults and complement measured counters.

C - Incorrect: Rule configuration may follow findings, but is not the act or direct purpose of log analysis.

D - Incorrect: Encryption protects communications; parsing historical records does not itself encrypt traffic.

Primary note proof

Study-note section NSA35

Count the event actually requested, use timestamps/hosts to correlate, and separate absence of matches from failure to read the log. Logs support security and performance analysis, so neither purpose is universally exclusive.

Study-note section NSA22

Measure a baseline, sample over an interval, compare with workload and correlate symptoms before changing hardware. CPU utilization/run queues identify processing pressure; low free memory alone is weak evidence because cache may be reclaimable. Persistent paging with high latency indicates memory pressure. Storage capacity and I/O latency are separate: a disk can be nearly empty but saturated, or nearly full while lightly loaded.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

System Monitoring Software Essentials
In order to keep your system fit for purpose, your monitoring activities need to cover the following priorities:
• Acceptable delivery speeds
• Constant availability
• Preventative maintenance
• Software version monitoring and patching
• Intrusion detection
• Data integrity
• Security monitoring
• Attack mitigation
• Virus prevention and detection

Data Communication and Computer Networking module | PDF page 54 | lines 49-63

• Monitoring a range of server metrics: availability, CPU usage, memory usage, disk space, response time, and upload/download rates. Monitoring CPU temperature and power supply voltages.
• Monitoring applications. Using deep knowledge of common applications and services to monitor key server processes, including web servers, database servers, and application stacks.
• Automatically alerting you of problems, such as servers or network devices that are overloaded or down, or worrisome trends. Customized alerts that can use multiple methods to contact you - email, SMS text messages, pager, etc.

• Triggering actions in response to alerts, to handle certain classes of problems automatically.

Question sources

Computer Science MCQ practice collection · Question 605

Question 78 | AM0606

A client has valid IP configuration. Which listed utility can send ICMP echo requests to a gateway as the next reachability diagnostic?

A. Ping

B. VPN

C. FTP

D. DNS

Hint

Match the requested observation to the tool’s actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer A: Ping

Ping sends ICMP echo requests and checks the returned replies. Testing the gateway can help isolate whether the client can exchange IP traffic with that next-hop device. A VPN provides extended protected connectivity, FTP transfers files and DNS resolves names; none is the requested echo-probe utility. Missing replies need further investigation because filtering or loss can suppress otherwise reachable devices.

Why each choice is right or wrong

A - Correct: Ping generates the requested ICMP echo probes.

B - Incorrect: A VPN provides protected connectivity rather than being an echo-test utility.

C - Incorrect: FTP transfers files rather than issuing the specified reachability probe.

D - Incorrect: DNS resolves names rather than sending ICMP echo requests.

Primary note proof

Study-note section NSA-S04

Ping uses ICMP echo messages to test IP reachability. A reply shows that the addressed host and path support that exchange; no reply alone does not prove the host is disconnected, because filtering or loss can suppress responses. Traceroute instead discovers responding intermediate hops through packet-lifetime expiry; it is a path-discovery diagnostic, not the same question as an end-to-end echo test.

Study-note section NSA10

DNS resolves names; DHCP supplies host configuration. SMTP sends/relays mail, while IMAP/POP retrieve it. FTP is a named file-transfer protocol; HTTP also transfers files, so “transfers files” is not an exclusive FTP description. HTTP/HTTPS serve web resources. SSH provides encrypted remote login; SFTP transfers files over SSH and differs from FTP or FTPS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 27 | lines 26-28

Ping (Packet Internet Groper) uses ICMP echo
messages to check the physical and logical
connectivity of machineson a network.

Data Communication and Computer Networking module | PDF page 28 | lines 37-45

The Process/Application Layer Protocols
• Telnet
• File Transfer Protocol (FTP)
• Network File System (NFS).
• Simple Mail Transfer Protocol (SMTP
• POP3 is used to receive mail.
• Simple Network Management Protocol (SNMP)
• Domain Name Service (DNS)
• Dynamic Host Configuration Protocol (DHCP)

Question sources

Computer Science MCQ practice collection · Question 606

Question 79 | AM0607

An administrator is monitoring network capacity and utilization, rather than conducting a vulnerability assessment. Which activity directly measures that objective?

A. Identifying security vulnerabilities

B. Tracking network bandwidth usage

C. Configuring user access controls

D. Performing system backups

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer B: Tracking network bandwidth usage

Capacity monitoring compares observed traffic with the available link rate and follows usage over time. Tracking bandwidth usage therefore measures the specified performance objective. Vulnerability assessment searches for weaknesses, access-control configuration changes permissions, and backups preserve data for recovery. All can be administration activities, but they do not measure the traffic utilization asked for here.

Why each choice is right or wrong

A - Incorrect: Vulnerability identification assesses weaknesses rather than measuring capacity utilization.

B - Correct: Bandwidth-usage tracking measures traffic utilization against available capacity.

C - Incorrect: Configuring access controls changes authorization rather than measuring traffic load.

D - Incorrect: Backups preserve recovery copies rather than observing utilization.

Primary note proof

Study-note section NSA22

> Measure interface bytes/second, errors, drops, packet delay/loss and utilization against link rate. Distinguish capacity (maximum available rate) from observed throughput. Logs can reveal both security incidents and performance faults. Performance counters expose measured events; system calls/APIs and log parsers are ways scripts can collect them.

Study-note section NSA23

> An IDS detects/alerts; an IPS can block inline. Scanning identifies potential weaknesses rather than proving successful exploitation. A captured packet, open port or alert needs context. Monitoring network usage is not the same as configuring access control or encrypting communications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 29 | lines 44-56

> Scope of Network administration
> The management of a network, usually called network administration, consists of procedures and services that keep the network running properly. An important part of network management entails making sure that the network is available (or up and running as IT professionals say) when employees and managers need it.Other admin activities are:
> • Monitoring the network capacity to ensure that alltransmission requirements can be met.
> • Adding capacity to the network by increasing band

width interconnecting additional modes, or
creating and interconnecting additional networks.

Data Communication and Computer Networking module | PDF page 54 | lines 13-17

Bandwidth monitoring tools and NetFlow and sFlow
based traffic analyzers help you stay aware of the activity,
capacity, and health of your network. They allow you to
watch traffic as it flows through routers and switches, or
arrive at and leaves hosts.

Question sources

Computer Science MCQ practice collection · Question 607

Question 80 | AM0608

Which troubleshooting approach tests a middle layer, then narrows the working/failing boundary upward or downward?

A. Divide and conquer

B. Trial and error

C. Random testing

D. Full system restore

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: Divide and conquer

Divide-and-conquer starts with a test at a middle layer and uses the result to decide which side of the boundary needs further investigation. It therefore narrows the scope systematically. Trial-and-error is a broader description of making attempts and does not specifically name this layer-boundary method. Random testing lacks the defined decision process, and a full restore is a recovery action rather than the approach described.

Why each choice is right or wrong

A - Correct: Divide-and-conquer is explicitly supported and isolates a working/failing boundary by testing a middle layer.

B - Incorrect: Trial-and-error does not specifically identify the middle-layer boundary-isolation approach described.

C - Incorrect: Random testing lacks the testable hypothesis and controlled isolation recommended by NSA08.

D - Incorrect: A full restore is a potentially disruptive recovery action, not the routine diagnostic approach identified in NSA08.

Primary note proof

Study-note section NSA08

> Use bottom-up checks from physical link, top-down checks from application errors, or divide-and-conquer by testing a middle layer. Divide-and-conquer narrows the working/failing boundary; it is not random trial-and-error. Rebooting, updates, resetting settings and scanning can all be troubleshooting actions in suitable circumstances; use fault evidence to choose among them.

Study-note section NSA08

> Define the symptom, scope, time and recent changes; gather the configuration, baseline and logs. Form a testable hypothesis, test the least disruptive explanation, isolate the failing component, apply a targeted fix, verify the user service and document the result. Change one relevant factor at a time when possible. A reboot can hide evidence and is not a diagnosis by itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 53 | lines 29-43

> In networking, it is typical to think of a system as a layered stack. User software sits at the top of the stack and system applications and services on the next layer down. Beneath the services and applications, you will encounter operating systems and firmware. Theperformance of software elements needs to be monitoredas an application stack.
> Users will notice performance problems with the software that they use, but those problems rarely arise within that software. All layers of the application stack need to be examined to find the root cause of performance issues. You need to head off problems with real-time statusmonitoring before they occur. Monitoring tools help you spot errors and service failures before they start to impactusers.

Data Communication and Computer Networking module | PDF page 51 | lines 2-10

> monitoring is normally done as the first and last steps of a three-step process:
> Monitoring to identify the nature and scope of the resource shortages that are causing the performance problems.
> The data produced from monitoring is analyzed and a course of action (normally performance tuning

and/or the procurement of additional hardware) is taken to resolve the problem

Question sources

Computer Science MCQ practice collection · Question 608

Question 81 | AM0609

A static network diagram records devices, links and their arrangement for later troubleshooting. What direct purpose does creating this diagram serve?

A. Documenting network configurations

B. Identifying security vulnerabilities

C. Monitoring network performance

D. Configuring network devices

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: Documenting network configurations

A network diagram documents the arrangement of devices and links and may record configuration relationships. That record helps a troubleshooter reason about expected connectivity. An analyst might subsequently infer weaknesses from it, but creating a static diagram does not itself establish vulnerabilities, sample live performance or apply device settings. The question asks for the direct documentation purpose.

Why each choice is right or wrong

A - Correct: Documenting the network arrangement and configuration is the direct purpose.

B - Incorrect: A diagram can inform security analysis, but creating it does not itself establish vulnerabilities.

C - Incorrect: A static diagram does not measure live performance counters or traffic.

D - Incorrect: Drawing the topology does not apply settings to devices.

Primary note proof

Study-note section NSA02

Switching normally forwards frames using MAC addresses within a LAN; routing forwards IP packets between networks. TCP supplies an ordered reliable byte stream, while UDP preserves datagrams without that guarantee. Ports distinguish application endpoints, not physical connectors. Draw switches, links, subnets, gateways, service hosts and dependencies in a network diagram. A topology's shape alone does not guarantee security.

Study-note section NSA01

Use documentation, standardization, least privilege, automation and measured change. Record the current state, intended change, rollback and validation. Schedule maintenance according to service impact and urgency, not merely because a calendar date arrived. Fault management restores failures; performance management measures service behavior; configuration management tracks settings; security management controls access; accounting tracks usage.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 19-33

Although the specifics of being a system administrator may change from platform to platform, there are underlying themes that do not. These themes make up the philosophy of system administration. The themes are:
• Automate everything
• Document everything
• Communicate as much as possible
• Know your resources
• Know your users
• Know your business
• Security cannot be an afterthought
• Plan ahead
• Expect the unexpected
• Backup and disaster recovery planning
• Patching

Data Communication and Computer Networking module | PDF page 22 | lines 45-52

Inter-connectivity: components of a network, including end devices and interconnecting devices, can be connected to each other differently in some fashion. By connectedness we mean either logically, physically or both ways. Network topology refers to the shape of a network, or the network‘s layout. It is the geometric representation of the relationship of all the links and linking devices to one another. There are

Question sources

Computer Science MCQ practice collection · Question 609

Question 82 | AM0610

Which measures can provide a network security control under their ordinary stated purposes? Select all that apply.

A. Intrusion Detection System (IDS)

B. User authentication

C. Data encryption

D. Network traffic analysis

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer A, B, C, D: Intrusion Detection System (IDS); User authentication; Data encryption; Network traffic analysis

An IDS detects suspicious activity and alerts, while traffic analysis can provide additional detective evidence. User authentication establishes identity for subsequent access decisions. Correctly implemented encryption protects data against unauthorized disclosure. These mechanisms address different security needs, so all are examples of controls under the broad question. Their inclusion does not imply that any single mechanism provides complete security by itself.

Why each choice is right or wrong

A - Correct: An IDS supplies detective monitoring and alerts about suspicious activity.

B - Correct: Authentication establishes an identity before access decisions and supports controlled admission.

C - Correct: Appropriate encryption protects transmitted content against unauthorized disclosure.

D - Correct: Traffic analysis can detect suspicious behavior and is explicitly included among security controls.

Primary note proof

Study-note section NSA23

> An IDS detects/alerts; an IPS can block inline. Scanning identifies potential weaknesses rather than proving successful exploitation. A captured packet, open port or alert needs context. Monitoring network usage is not the same as configuring access control or encrypting communications.

Study-note section NSA28

> Encryption provides confidentiality with correct key/nonce use; authenticated encryption adds integrity/authenticity. Unspecified encryption alone does not guarantee both properties. A password hash supports verification without recovering the original password; it is not a reversible credential store. Encryption can protect a recoverable credential only if its key is separately protected.

Study-note section NSA30

Segmentation separates groups of systems and enforces allowed flows with routing/filtering policies; VLAN membership alone does not establish complete security. NAC assesses device/user admission and can reject or quarantine unauthorized/noncompliant devices. Authentication proves an identity, while access rules decide which resources it may use.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 54 | lines 18-30

System Monitoring Software Essentials
In order to keep your system fit for purpose, your monitoring activities need to cover the following priorities:
• Acceptable delivery speeds
• Constant availability
• Preventative maintenance
• Software version monitoring and patching
• Intrusion detection
• Data integrity
• Security monitoring
• Attack mitigation
• Virus prevention and detection

Data Communication and Computer Networking module | PDF page 54 | lines 13-17

Bandwidth monitoring tools and NetFlow and sFlow based traffic analyzers help you stay aware of the activity, capacity, and health of your network. They allow you to watch traffic as it flows through routers and switches, or arrive at and leaves hosts.

Data Communication and Computer Networking module | PDF page 16 | lines 33-42

Presentation Layer
• The communicating devices may be having different platforms. The presentation layer performs translation, encryption and compression of data.
• The presentation layer receives the data from the

application layer adds header which contains information related to encryption and compression and sends it to the session layer.
• So, the main responsibility of the Presentation layer is translation, compression and encryption.

Question sources

Computer Science MCQ practice collection · Question 610

Question 83 | AM0611

What is the direct network-structural purpose of segmentation with enforced intersegment policies? Select the most specific description.

A. Controlling user access to resources

B. Encrypting network communications

C. Isolating network segments for security purposes

D. Monitoring network traffic

Hint

Identify whether the task requires input/output, control flow, code organization, process execution or collection of a specific measurement.

Answer C: Isolating network segments for security purposes

Segmentation separates groups of systems and enforces permitted flows between them. This structural isolation can support access restrictions, but controlling user access is a broader result also achieved through other mechanisms. The most specific description is therefore isolating network segments for security. Encryption and monitoring are distinct controls and do not automatically follow from dividing the network into segments.

Why each choice is right or wrong

A - Incorrect: Access control may be an outcome, but this broad description does not name segmentation’s direct structural purpose.

B - Incorrect: Segmentation restricts connectivity but does not inherently encrypt the communications it permits.

C - Correct: Segmentation isolates system groups into network segments and enforces intersegment flows.

D - Incorrect: Traffic monitoring observes activity; it is distinct from establishing and enforcing segment boundaries.

Primary note proof

Study-note section NSA30

Segmentation separates groups of systems and enforces allowed flows with routing/filtering policies; VLAN membership alone does not establish complete security. NAC assesses device/user admission and can reject or quarantine unauthorized/noncompliant devices. Authentication proves an identity, while access rules decide which resources it may use.

Study-note section NSA30

Use authenticated protected remote administration such as SSH or properly secured management services, with restricted source access and auditing. VPNs provide protected connectivity under their defined security model; they are not packet analyzers. Firewall filtering permits/blocks traffic according to policy; it is not inherently encryption. Losing remote connectivity while changing a management route/firewall is an operational risk to plan for.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

• Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 22 | lines 45-52

Inter-connectivity: components of a network, including end devices and interconnecting devices, can be connected to each other differently in some fashion. By connectedness we mean either logically, physically or both ways. Network topology refers to the shape of a network, or the network‘s layout. It is the geometric representation of the relationship of all the links and linking devices to one another. There are

Question sources

Computer Science MCQ practice collection · Question 611

Question 84 | AM0612

What is the role of a Network Access Control (NAC) system in network security?

A. Managing user account credentials

B. Monitoring network bandwidth

C. Detecting and blocking unauthorized devices

D. Auditing network activity

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer C: Assess admission and reject unauthorized devices

NSA30 defines Network Access Control as assessing device/user admission and rejecting or quarantining unauthorized or noncompliant devices. C matches that admission function. Credential management, bandwidth monitoring and general activity auditing can support network operations, but they do not state the specific NAC enforcement role.

Why each choice is right or wrong

A - Incorrect: Managing account credentials is identity administration, not the complete device-admission role defined for NAC.

B - Incorrect: Bandwidth monitoring measures network use rather than deciding whether a device may join.

C - Correct: NAC can detect unauthorized/noncompliant devices during admission and reject or quarantine them.

D - Incorrect: Auditing records activity; NAC's distinguishing role is admission assessment and enforcement.

Primary note proof

Study-note section NSA30

> Segmentation separates groups of systems and enforces allowed flows with routing/filtering policies; VLAN membership alone does not establish complete security. NAC assesses device/user admission and can reject or quarantine unauthorized/noncompliant devices. Authentication proves an identity, while access rules decide which resources it may use.

Study-note section NSA22

> Measure interface bytes/second, errors, drops, packet delay/loss and utilization against link rate. Distinguish capacity (maximum available rate) from observed throughput. Logs can reveal both security incidents and performance faults. Performance counters expose measured events; system calls/APIs and log parsers are ways scripts can collect them.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

> • Putting security procedures in place to make certain that only authorized users have access to the network and ensuring that all security procedures are followed
> • Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Data Communication and Computer Networking module | PDF page 34 | lines 7-14

> Domain Controllers
> A domain controller (DC) is a server computer that responds to security authentication requests within a computer network domain. It is a network server that is responsible for allowing end devices to access shared domain resources. It authenticates users, stores user account information and enforces security policy for a domain. It is most commonly implemented in Microsoft

Question sources

Computer Science MCQ practice collection · Question 612

Question 85 | AM0613

In the note’s bottom-up connectivity procedure, which listed action belongs to the initial physical-link check?

A. Checking physical cable connections

B. Updating antivirus software

C. Configuring firewall rules

D. Performing system backups

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A: Checking physical cable connections

A bottom-up procedure begins with power, cable or connector condition and link state before examining interfaces, addressing, routes or application policy. Checking physical cable connections directly fits that initial stage. Firewall configuration may become relevant later when evidence identifies filtering. Antivirus updates and backups serve maintenance or recovery purposes rather than the first physical-link inspection specified in this question.

Why each choice is right or wrong

A - Correct: Cable/connector and link-state checks directly diagnose physical connectivity problems.

B - Incorrect: Updating antivirus is maintenance rather than a direct connectivity-isolation technique supported by this stem.

C - Incorrect: Firewall rules may require later inspection, but are not the initial physical-link check.

D - Incorrect: Backups support recovery but do not directly test or correct the network-connectivity path.

Primary note proof

Study-note section NSA09

Check power, cable/connector and link state, then NIC/driver state, VLAN membership, address/prefix, local gateway, route, DNS and finally application port/authentication. A recognized NIC makes a missing driver less likely, not proof of a working physical link. Ethernet commonly uses RJ45 connectors; an RJ11 telephone jack is not the expected Ethernet connection.

Study-note section NSA08

Use bottom-up checks from physical link, top-down checks from application errors, or divide-and-conquer by testing a middle layer. Divide-and-conquer narrows the working/failing boundary; it is not random trial-and-error. Rebooting, updates, resetting settings and scanning can all be troubleshooting actions in suitable circumstances; use fault evidence to choose among them.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 16-18

• Diagnosing and troubleshooting problems on the network and determining the best course of action to take to solve them.

Data Communication and Computer Networking module | PDF page 48 | lines 29-40

Network Address Translation (NAT)
NAT translates the IP addresses of computers in a local network to a single IP address. This address is often used by the router that connects the computers to the Internet. The router can be connected to a DSL modem, cable modem, T1 line, or even a dial-up modem. When other computers on the Internet attempt to access computers within the local network, they only see the IP address of the router. This adds an extra level of security, since the router can be configured as a firewall, only allowing authorized systems to access the computers within the network.

Data Communication and Computer Networking module | PDF page 22 | lines 45-52

Inter-connectivity: components of a network, including end devices and interconnecting devices, can be connected to each other differently in some fashion. By connectedness we mean either logically, physically or both ways. Network topology refers to the shape of a network, or the network‘s layout. It is the geometric representation of the relationship of all the links and linking devices to one another. There are

Question sources

Computer Science MCQ practice collection · Question 613

Question 86 | AM0614

Which of the following scripting languages is commonly used for system administration tasks?

A. Java

B. C++

C. Python

D. HTML

Hint

Identify whether the task requires input/output, control flow, code organization, process execution or collection of a specific measurement.

Answer C: Python

NSA31 explicitly names Python among common system-administration scripting choices, along with shells and PowerShell. HTML is markup. Java and C++ are programming languages but are not the common scripting choice identified among these options. The answer concerns ordinary administration automation, not whether those languages could ever perform administrative work.

Why each choice is right or wrong

A - Incorrect: Java can implement programs, but it is not the common administration scripting choice identified in NSA31.

B - Incorrect: C++ is a compiled programming language rather than the listed common administration scripting choice.

C - Correct: Python is explicitly identified as a common language for administrative scripts.

D - Incorrect: HTML is markup and does not provide the administrative scripting control flow requested.

Primary note proof

Study-note section NSA31

> Scripts automate repeatable administration; they are not inherently faster than compiled programs or automatically secure. Python, shells and PowerShell are common choices; HTML is markup. Define inputs and outputs, use variables for state, if/elif/else for decisions, loops for repeated work and functions for reusable operations. Python indentation defines blocks; == compares while = assigns.

Study-note section NSA-S01

Least privilege limits an account or process to the rights its task needs. Agile development instead organizes frequent delivery and adaptation of software. Object-oriented programming organizes software around objects with state and behavior. Cloud computing supplies on-demand access to a shared pool of configurable computing resources. A development approach or hosting model does not itself define an access-rights principle.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 61 | lines 26-36

What is Unix/Linux?
The Unix OS is a set of programs that act as a link between the computer and the user. The computer program that allocates the system resources and coordinates all the details of the computer‘s internals is called the operating system or the kernel. Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

Question sources

Computer Science MCQ practice collection · Question 614

Question 87 | AM0615

What is the primary advantage of using scripts for system administration tasks?

A. Faster execution compared to compiled programs

B. Ability to automate repetitive tasks

C. Greater control over system resources

D. Enhanced security and encryption features

Hint

Identify whether the task requires input/output, control flow, code organization, process execution or collection of a specific measurement.

Answer B: Automate repetitive work

NSA31 identifies repeatable automation as the reason to use administration scripts. Their value is consistent execution of recurring tasks, not guaranteed speed relative to compiled code, stronger encryption or inherently greater resource control. Those properties depend on implementation and environment rather than following from the use of a script.

Why each choice is right or wrong

A - Incorrect: Scripts are not inherently faster than compiled programs; the note explicitly rejects that universal claim.

B - Correct: Automation of repetitive administrative work is the primary script benefit given in NSA31.

C - Incorrect: Greater resource control is not an inherent consequence of scripting rather than another implementation method.

D - Incorrect: Scripts are not automatically secure and do not inherently add encryption features.

Primary note proof

Study-note section NSA31

> Scripts automate repeatable administration; they are not inherently faster than compiled programs or automatically secure. Python, shells and PowerShell are common choices; HTML is markup. Define inputs and outputs, use variables for state, if/elif/else for decisions, loops for repeated work and functions for reusable operations. Python indentation defines blocks; == compares while = assigns.

Study-note section NSA31

> Make reruns predictable: compute the desired state, compare with current state and change only what differs. Validate a small dry run before mutation. Avoid requiring privilege for read-only reporting. A good script reports failures and has a meaningful exit status rather than printing success unconditionally.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 19-33

> Although the specifics of being a system administrator
> may change from platform to platform, there are
> underlying themes that do not. These themes make up the
> philosophy of system administration. The themes are:
> • Automate everything
> • Document everything
> • Communicate as much as possible
> • Know your resources
> • Know your users
> • Know your business
> • Security cannot be an afterthought
> • Plan ahead
> • Expect the unexpected
> • Backup and disaster recovery planning
> • Patching

Data Communication and Computer Networking module | PDF page 61 | lines 26-36

What is Unix/Linux?
The Unix OS is a set of programs that act as a link between the computer and the user. The computer program that allocates the system resources and coordinates all the details of the computer‘s internals is called the operating system or the kernel. Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

Question sources

Computer Science MCQ practice collection · Question 615

Question 88 | AM0616

In Python, which listed module provides an API for launching child programs and collecting their results?

A. echo

B. ifconfig

C. grep

D. subprocess

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: subprocess

Python’s subprocess module launches child programs and provides interfaces for collecting output, checking status and limiting runtime. It is a language module rather than a universal shell command. Echo prints values, ifconfig handles interface configuration, and grep searches text. Naming the language and asking for the module makes the intended process-execution abstraction precise without confusing it with one particular program a script might launch.

Why each choice is right or wrong

A - Incorrect: Echo prints values and is not a general child-process execution abstraction.

B - Incorrect: Ifconfig reports/configures network interfaces rather than generally launching arbitrary programs.

C - Incorrect: Grep searches text and is not the general program-execution mechanism requested.

D - Correct: The Python subprocess module launches child programs and returns their results.

Primary note proof

Study-note section NSA32

Python subprocess is a module, not a universal shell command. run launches a child program and returns its result; capture_output collects stdout/stderr, text=True decodes text, timeout bounds waiting and check=True raises for nonzero exit. Pass an argument list to avoid accidental shell interpretation; quote handling differs from composing a shell string.

Study-note section NSA32

Echo prints, grep searches text and ipconfig/ifconfig reports/configures networking; none is the general Python child-process abstraction. Specify the scripting language before selecting subprocess for process execution.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 61 | lines 26-36

What is Unix/Linux?
The Unix OS is a set of programs that act as a link between the computer and the user. The computer program that allocates the system resources and coordinates all the details of the computer‘s internals is called the operating system or the kernel. Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

Data Communication and Computer Networking module | PDF page 65 | lines 39-47

Network configuration commands
The ip Command
The distribution-specific scripts and graphical tools are front-ends to ip (or ifconfig and route on older systems) to display and configure the kernel‘s networking configuration. The ip command is used for assigning IP addresses to interfaces, for setting up routes to the Internet and to other networks, for displaying TCP/IP

Question sources

Computer Science MCQ practice collection · Question 616

Question 89 | AM0617

Which scripting feature is useful for handling errors and exceptions in system administration scripts?

A. Looping constructs

B. Conditional statements

C. Exception handling

D. Function definitions

Hint

Identify whether the task requires input/output, control flow, code organization, process execution or collection of a specific measurement.

Answer C: Exception handling

The NSA34 example uses try with specific exception handlers to report expected read failures while exposing unexpected programming errors. Exception handling is therefore the language feature directly intended for handling errors and exceptions. Loops repeat work, conditions choose branches and functions package reusable operations; none alone supplies exception interception.

Why each choice is right or wrong

A - Incorrect: Loops repeat operations and do not themselves intercept raised exceptions.

B - Incorrect: Conditionals choose branches from ordinary conditions but do not replace exception handlers.

C - Correct: Exception handling catches specified failures and controls reporting or recovery when exceptions occur.

D - Incorrect: Functions organize reusable work; they still require appropriate error/exception handling inside or around calls.

Primary note proof

Study-note section NSA-S05

> Exception handling transfers control from a failing operation to a matching handler. A try block contains the attempted work; an except clause handles specified raised exceptions. A loop repeats operations, a conditional chooses an ordinary branch, and a function packages reusable work; these constructs do not by themselves intercept exceptions.

Study-note section NSA31

> Scripts automate repeatable administration; they are not inherently faster than compiled programs or automatically secure. Python, shells and PowerShell are common choices; HTML is markup. Define inputs and outputs, use variables for state, if/elif/else for decisions, loops for repeated work and functions for reusable operations. Python indentation defines blocks; == compares while = assigns.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 61 | lines 26-36

> What is Unix/Linux?
> The Unix OS is a set of programs that act as a link between the computer and the user. The computer program that allocates the system resources and coordinates all the details of the computer‘s internals is called the operating system or the kernel. Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

Question sources

Computer Science MCQ practice collection · Question 617

Question 90 | AM0618

Which concept directly reads configuration bytes from a file or writes updated content back to it, rather than interpreting or holding values?

A. File input/output

B. Regular expressions

C. String manipulation

D. Variable assignment

Hint

Identify whether the task requires input/output, control flow, code organization, process execution or collection of a specific measurement.

Answer A: File input/output

File input/output obtains configuration content from storage and persists revised content. Regular expressions and string operations can interpret or construct that content after it has been read, while variables hold values during processing. All can assist a configuration script, but only file I/O directly crosses the file-storage boundary asked about. Structured formats should still be parsed with their appropriate parsers.

Why each choice is right or wrong

A - Correct: File I/O obtains and persists the configuration content being manipulated.

B - Incorrect: Regex interprets patterns in content rather than directly reading or writing file storage.

C - Incorrect: String manipulation processes text rather than directly transferring it to or from a file.

D - Incorrect: Variable assignment holds values in program state rather than performing file I/O.

Primary note proof

Study-note section NSA33

> File I/O obtains/persists configuration; string operations parse values; regex can recognize patterns. File I/O, string operations and regex can all help manipulate configuration; choose according to the task. Use the format's parser for structured JSON/INI rather than blindly replacing every matching substring. Open text with an explicit encoding and use with so files close even when processing fails.

Study-note section NSA31

> Scripts automate repeatable administration; they are not inherently faster than compiled programs or automatically secure. Python, shells and PowerShell are common choices; HTML is markup. Define inputs and outputs, use variables for state, if/elif/else for decisions, loops for repeated work and functions for reusable operations. Python indentation defines blocks; == compares while = assigns.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 65 | lines 39-47

> Network configuration commands
> The ip Command
> The distribution-specific scripts and graphical tools
> are front-ends to ip (or ifconfig and route on older
> systems) to display and configure the kernel‘s
> networking configuration. The ip command is used for
> assigning IP addresses to interfaces, for setting up routes
> to the Internet and to other networks, for displaying
> TCP/IP

Data Communication and Computer Networking module | PDF page 64 | lines 28-31

> /var: Storage for all variable and temporary files
> created by users, such as log files, temporary files
> downloaded from the Internet, or to keep an image of a
> CD before burning it.

Question sources

Computer Science MCQ practice collection · Question 618

Question 91 | AM0619

Which scripting technique recognizes textual patterns and extracts matching fields or events from individual system-log lines?

A. Database integration

B. File compression

C. Log rotation

D. Regular expression matching

Hint

Identify whether the task requires input/output, control flow, code organization, process execution or collection of a specific measurement.

Answer D: Regular expression matching

Regular expressions describe text patterns and can capture fields from matching log lines. A script can then count events, correlate entries or report anomalies. Database integration supplies storage or query infrastructure, compression reduces data size, and rotation manages log segments and retention. Those may support an overall monitoring system, but they do not themselves perform the specific textual matching operation requested.

Why each choice is right or wrong

A - Incorrect: Database integration stores or queries information rather than specifying the requested text pattern.

B - Incorrect: Compression reduces data size rather than extracting matching fields.

C - Incorrect: Rotation manages file size and historical segments, not textual matching.

D - Correct: Regular expressions match patterns and capture requested fields or events.

Primary note proof

Study-note section NSA35

> Regex recognizes textual patterns; it does not encrypt, compress or rotate logs. search finds a match anywhere, while fullmatch requires the entire string to fit. Use raw Python strings for backslash-heavy patterns, capture only the needed fields and treat unmatched/malformed lines explicitly. Log rotation bounds file size and preserves historical segments; it is complementary to parsing.

Study-note section NSA35

> Count the event actually requested, use timestamps/hosts to correlate, and separate absence of matches from failure to read the log. Logs support security and performance analysis, so neither purpose is universally exclusive.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 64 | lines 14-17

/proc: Virtual file system containing system resources information. You can type man proc command on terminal to see more information about the meaning of the files in proc.

Data Communication and Computer Networking module | PDF page 54 | lines 49-63

• Monitoring a range of server metrics: availability, CPU usage, memory usage, disk space, response time, and upload/download rates. Monitoring CPU temperature and power supply voltages.
• Monitoring applications. Using deep knowledge of common applications and services to monitor key server processes, including web servers, database servers, and application stacks.
• Automatically alerting you of problems, such as servers or network devices that are overloaded or down, or worrisome trends. Customized alerts that can use multiple methods to contact you - email, SMS text messages, pager, etc.
• Triggering actions in response to alerts, to handle certain classes of problems automatically.

Data Communication and Computer Networking module | PDF page 64 | lines 28-31

/var: Storage for all variable and temporary files created by users, such as log files, temporary files downloaded from the Internet, or to keep an image of a CD before burning it.

Question sources

Computer Science MCQ practice collection · Question 619

Question 92 | AM0620

Which scripting language feature allows passing command-line arguments to a script?

A. Functions

B. Modules

C. Classes

D. Command-line parsing

Hint

Match the requested observation to the tool’s actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer D: Command-line parsing

NSA34 identifies command-line parsing as the mechanism for interpreting invocation arguments as named, typed inputs and options. The example uses argparse. Functions, modules and classes can organize a parser's implementation, but they do not by themselves define argument interpretation or validation; D states the requested feature directly.

Why each choice is right or wrong

A - Incorrect: Functions organize callable work but do not inherently parse invocation arguments.

B - Incorrect: Modules organize code; a parser module still needs command-line parsing behavior to interpret inputs.

C - Incorrect: Classes organize state/behavior and do not themselves define command-line argument processing.

D - Correct: Command-line parsing interprets the arguments and options supplied when invoking a script.

Primary note proof

Study-note section NSA34

> Command-line parsing converts user inputs into named, typed values and displays usage. In Python argparse, positional arguments supply required inputs and flags enable options. Validate ranges and paths before work; do not let unchecked input become executable shell text.

Study-note section NSA-S05

> A module groups definitions and statements that can be imported. A class bundles data with functionality for its objects. These organize a program, while command-line parsing interprets the arguments and flags supplied at invocation. Their implementation can use functions or classes, but organization alone does not parse arguments.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 61 | lines 26-36

> What is Unix/Linux?
> The Unix OS is a set of programs that act as a link between the computer and the user. The computer program that allocates the system resources and coordinates all the details of the computer‘s internals is called the operating system or the kernel. Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

Question sources

Computer Science MCQ practice collection · Question 620

Question 93 | AM0621

Which scripting technique is commonly used for automating system backup and restore tasks?

A. File compression

B. Version control

C. Scheduling

D. Event logging

Hint

Identify whether the task requires input/output, control flow, code organization, process execution or collection of a specific measurement.

Answer C: Scheduling

Automating recurring backup/restore work requires choosing when the operation runs. NSA37 assigns this to schedulers such as cron/systemd timers or Windows Task Scheduler. Compression packages data, version control tracks script changes and logging records outcomes. Those can support a backup process but do not themselves trigger the scheduled task.

Why each choice is right or wrong

A - Incorrect: Compression packages data but does not determine when a backup or restore operation executes.

B - Incorrect: Version control tracks changes in scripts and does not replace scheduling or data backups.

C - Correct: Scheduling triggers recurring work at configured times through the platform's scheduler.

D - Incorrect: Event logging records execution/results rather than initiating the recurring operation.

Primary note proof

Study-note section NSA37

> Schedule recurring work using cron/systemd timers on suitable Unix systems or Task Scheduler on Windows. A user crontab uses minute,hour,day-of-month,month,day-of-week then command. Use absolute paths, known environment, protected inputs, logs, bounded runtime and a lock when overlap is unsafe. Do not assume an interactive PATH, current directory or stored credentials exist in a scheduled job.

Study-note section NSA37

Compression packages data; scheduling chooses when the operation runs. Test exit status and the archive contents, then restore a sample into an isolated destination. A successful command alone does not prove a usable backup. Version control of scripts helps review but does not replace data backup.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 6-8

• Backing up the network software and data
regularly to protect against the failure of network
or any of its components

Data Communication and Computer Networking module | PDF page 30 | lines 19-33

Although the specifics of being a system administrator
may change from platform to platform, there are
underlying themes that do not. These themes make up the
philosophy of system administration. The themes are:
• Automate everything
• Document everything
• Communicate as much as possible
• Know your resources
• Know your users
• Know your business
• Security cannot be an afterthought
• Plan ahead
• Expect the unexpected
• Backup and disaster recovery planning
• Patching

Question sources

Computer Science MCQ practice collection · Question 621

Question 94 | AM0622

A script must recover a stored service credential to authenticate later. With the decryption key protected separately, which listed mechanism cryptographically protects that recoverable credential at rest?

A. Environment variables

B. Encryption algorithms

C. Hashing functions

D. Configuration files

Hint

Identify whether the task requires input/output, control flow, code organization, process execution or collection of a specific measurement.

Answer B: Encryption algorithms

The script needs the actual credential for later authentication, so a one-way password hash cannot meet the recoverability requirement. Encryption can protect the stored credential while allowing authorized decryption, provided the key is separately protected. Environment variables and configuration files describe storage or delivery choices and are not automatically encrypted. The explicit use case therefore identifies encryption without treating it as sufficient when its key is exposed.

Why each choice is right or wrong

A - Incorrect: Environment variables deliver or hold values but do not by themselves encrypt stored credentials.

B - Correct: Encryption protects a recoverable credential when its key is separately protected.

C - Incorrect: A one-way hash supports verification but cannot recover the service credential.

D - Incorrect: A configuration file is a storage choice and is not automatically cryptographically protected.

Primary note proof

Study-note section NSA28

> Encryption provides confidentiality with correct key/nonce use; authenticated encryption adds integrity/authenticity. Unspecified encryption alone does not guarantee both properties. A password hash supports verification without recovering the original password; it is not a reversible credential store. Encryption can protect a recoverable credential only if its key is separately protected.

Study-note section NSA28

> Keep reusable secrets out of source code and routine logs; obtain them from an appropriately protected credential service or restricted runtime source. Environment variables and configuration files are transport/storage choices, not automatic encryption. A script that must authenticate to another service often needs the actual credential/token, so replacing it with a one-way hash will not work. State whether the question is about password verification or recoverable service credentials.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 30 | lines 9-15

> • Putting security procedures in place to make
> certain that only authorized users have access to
> the network and ensuring that all security
> procedures are followed

• Making sure the network personnel can respond quickly and effectively in the event of a network operational or security failure.

Question sources

Computer Science MCQ practice collection · Question 622

Question 95 | AM0623

Which techniques can collect CPU or memory measurements when the relevant logs or OS interfaces expose those metrics? Select all that apply.

A. Log parsing

B. System calls

C. Signal handling

D. Performance counters

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A, B, D: Log parsing; System calls; Performance counters

Resource measurements can be recorded in logs, exposed through operating-system APIs or published as performance counters. Scripts may use any of those interfaces when the requested metrics are available. Signal handling instead reacts to process notifications and is not itself a CPU or memory measurement interface. The multiple-selection instruction preserves the complementary collection techniques without claiming one universal platform-independent API.

Why each choice is right or wrong

A - Correct: Parsing logs can collect recorded resource metrics when suitable measurements are present.

B - Correct: OS system calls/APIs can expose resource measurements used by monitoring scripts.

C - Incorrect: Signal handling manages process notifications; it is not the measurement interface for CPU/memory utilization.

D - Correct: Performance counters directly expose measured resource events and utilization-related values.

Primary note proof

Study-note section NSA36

/proc/meminfo exposes MemAvailable, an estimate of memory available without swapping; MemFree alone excludes reclaimable memory. Windows Performance Counters and OS APIs expose analogous measurements through different interfaces. System calls, counters and log

parsing can all contribute; selecting one as the only monitoring technique is unjustified without a platform/API.

Study-note section NSA-S05

A signal handler responds when a process receives a specified asynchronous event notification. Handling such a notification is distinct from reading CPU or memory measurements through an OS interface, a performance counter or a log that actually records the metric.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 50 | lines 32-43

Before you can monitor resources, you first have to know what resources there are to monitor. All systems have the following resources available:
• CPU power
• Bandwidth
• Memory
• Storage
System Performance Monitoring
System performance monitoring is normally done in response to a performance problem. Either the system is running too slowly, or programs (and sometimes even the entire system) fail to run at all. In either case, performance

Data Communication and Computer Networking module | PDF page 47 | lines 2-5

registry. The kernel, device drivers, services, Security Accounts Manager, and user interface can all use the registry. The registry also allows access to counters for profiling system performance.

Data Communication and Computer Networking module | PDF page 64 | lines 14-17

/proc: Virtual file system containing system resources information. You can type man proc command on terminal to see more information about the meaning of the files in proc.

Question sources

Computer Science MCQ practice collection · Question 623

Question 96 | CS0188

Suppose you are a network administrator and want to determine the path between two connections, which often goes through multiple routers. Which command would you use?

A. ping

B. nslookup

C. traceroute

D. All of the above

Hint

Match the requested observation to the tool's actual output; distinguish configuration, name lookup, reachability and hop discovery.

Answer C: traceroute

Traceroute discovers responding hops toward a destination by causing packet-lifetime expiry at successive distances. It is therefore the appropriate command for investigating a multi-router path. Ping checks an end-to-end echo exchange and nslookup inspects DNS resolution, which do not supply the requested hop sequence. Some intermediate devices may not reply, so the displayed trace is evidence from responding hops rather than a guarantee that every router is visible.

Why each choice is right or wrong

A - Incorrect: Ping checks end-to-end reachability, not the requested hop sequence.

B - Incorrect: Nslookup queries DNS information, not intermediate router hops.

C - Correct: Traceroute discovers responding intermediate hops along the path.

D - Incorrect: The other commands do not all perform route-hop discovery.

Primary note proof

Study-note section NSA-S04

> Ping uses ICMP echo messages to test IP reachability. A reply shows that the addressed host and path support that exchange; no reply alone does not prove the host is disconnected, because filtering or loss can suppress responses. Traceroute instead discovers responding intermediate hops through packet-lifetime expiry; it is a path-discovery diagnostic, not the same question as an end-to-end echo test.

Study-note section NSA07

> netstat displays connections/endpoints/protocols and TCP states; netstat -ano displays numeric addresses/ports and PIDs. Use the exact command name netstat rather than netstart. ping probes IP reachability; traceroute/tracert reveals responding hops using TTL/hop-limit expiry. ICMP filtering can prevent replies without proving the service is down. nslookup inspects DNS.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 27 | lines 26-28

> Ping (Packet Internet Groper) uses ICMP echo messages to check the physical and logical connectivity of machineson a network.

Data Communication and Computer Networking module | PDF page 27 | lines 29-31

> Traceroute Using ICMP timeouts, Traceroute is used to discover the path a packet takes as it traverses an Internetwork.

Question sources

Computer Science Exit Exam practice collection · Question 188

Question 97 | CS0189

Which statements correctly describe IPv4 loopback? In D, 127.0.0.1 means the conventional example rather than the entire reserved range. Select all that apply.

A. It helps to refer to the host itself

B. It allows programs to access network protocols for local services

C. It is an external pseudo address

D. It is denoted by the IP address: 127.0.0.1

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer A, B, D: It helps to refer to the host itself; It allows programs to access network protocols for local services; It is denoted by the IP address: 127.0.0.1

Loopback lets a host address its own protocol stack and reach local services without an external link. The conventional IPv4 address is 127.0.0.1, within the broader reserved 127/8 block. It is therefore local self-reference rather than an external address. The example wording permits all three true descriptions without incorrectly restricting the entire loopback range to a single address.

Why each choice is right or wrong

A - Correct: Loopback refers to the host itself rather than a remote device.

B - Correct: Programs can reach local services through network protocols using loopback.

C - Incorrect: Loopback is local and does not require an external network link.

D - Correct: 127.0.0.1 is the conventional IPv4 loopback address, although the reserved block is broader.

Primary note proof

Study-note section NSA03

127.0.0.1 conventionally names the local IPv4 loopback interface; 127/8 is reserved for loopback. Loopback tests local protocol processing, not cable, switch or remote reachability. Programs can use it for local services. Several loopback descriptions or several 127/8 addresses may be valid; compare each with the stated role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Data Communication and Computer Networking module | PDF page 65 | lines 31-38

/etc/hosts
The /etc/hosts file always contains the localhost IP
address, 127.0.0.1, which is
usedfor interprocess communication.
Never remove this line! Sometimes contains
addresses of additional hosts, whichcan be contacted
without using an external naming service such as
DNS (the Domain Name Server).

Question sources

Computer Science Exit Exam practice collection · Question 189

Question 98 | NS0296

Which listed item is a traffic-prioritization mechanism that can operate across layers, rather than an application mediator or service installation/configuration activity?

A. Proxies and agents

B. Installing a new service

C. E-mail configuration

D. Quality of service

Hint

Identify the required operational function, then compare its scope, inputs and constraints with the cited note definitions.

Answer D: Quality of service

Quality of service classifies, queues and prioritizes traffic to meet performance objectives such as low delay or limited loss. It can span layers while supporting applications. Proxies and agents are mediating components, and installing a service or configuring email are administrative activities. The revised question uses those explicit functional categories, so it does not require an unsupported taxonomy that labels every item an application-layer service.

Why each choice is right or wrong

A - Incorrect: Proxies and agents mediate requests rather than naming the traffic-prioritization mechanism.

B - Incorrect: Installing a service is an administrative activity.

C - Incorrect: Email configuration is an administrative activity.

D - Correct: QoS classifies and prioritizes traffic and can operate across layers.

Primary note proof

Study-note section NSA12

> A proxy acts on behalf of a client or backend service and can cache, filter or mediate application requests. A default IP gateway is a next-hop router; an application gateway may translate protocols. It need not be a modem/router combination. These terms describe functions that can coexist on one machine.

Study-note section NSA12

> QoS classifies, queues and prioritizes traffic to meet performance objectives such as low delay or limited loss. It can operate across layers and support applications, so it is not exclusively an application-layer service. Installing a new service is an administrative action; email configuration is another action and proxies/agents are components. Distinguish service-management actions from components using a stated taxonomy.

Question sources

Computer Science Exit Exam · 2018 · Question 13

Question 99 | NS0313

A data-center volume has exactly two equal disks and must retain service after either disk fails. One disk’s usable capacity is sufficient. Which listed RAID level meets these requirements?

A. RAID level 3

B. RAID level1

C. RAID level 2

D. RAID level 0

Hint

Compare striping granularity, redundancy placement, usable capacity and required disk-failure tolerance.

Answer B: RAID level1

With exactly two equal disks, RAID1 mirrors the data and leaves one disk's usable capacity while tolerating either disk failing. RAID0 provides no disk-failure tolerance. RAID2 and RAID3 rely on different striping and redundancy layouts rather than the two-disk mirrored arrangement that meets these requirements. The choice follows the stated disk count, capacity and failure goal rather than the institution's size or industry.

Why each choice is right or wrong

A - Incorrect: RAID3 uses byte striping with dedicated parity, not the required two-disk fault-tolerant layout.

B - Correct: A two-disk RAID1 mirror provides one disk's capacity and survives either disk failing.

C - Incorrect: RAID2 uses bit striping and Hamming redundancy rather than the required two-disk solution.

D - Incorrect: RAID0 cannot tolerate a disk failure.

Primary note proof

Study-note section NSA25

> RAID0 stripes for capacity/throughput with no disk-failure tolerance. RAID1 mirrors; a two-disk mirror has one disk's usable capacity and survives one disk failure. RAID2 uses bit striping/Hamming-code redundancy, RAID3 byte striping with dedicated parity, RAID4 block striping with dedicated parity and RAID5 block striping with distributed parity. RAID6 distributes dual parity and tolerates two disk failures. RAID10 stripes mirrored pairs; survival depends on which disks fail.

Study-note section NSA25

> With N equal disks of capacity D, RAID0 usable=ND, paired RAID1/10≈ND/2, RAID5=(N−1)D and RAID6=(N−2)D. Hardware RAID may offload work to controller resources/cache but does not universally outperform software RAID. Rebuilds consume bandwidth; parity writes have overhead. A large university/bank alone does not select a level. Striping plus parity is not uniquely RAID5 unless distributed block parity is specified.

Question sources

Computer Science Exit Exam · 2018 · Question 37

Question 100 | NS0324

A storage administrator configures four equal 2 TB disks. Each stripe stores data blocks on three disks and a parity block on the fourth; the disk holding parity changes between stripes. Ignoring metadata, which listed RAID layout provides 6 TB of usable capacity and tolerates one disk failure?

A. RAID 2

B. RAID 1

C. RAID 5

D. RAID 0

Hint

Compare striping granularity, redundancy placement, usable capacity and required disk-failure tolerance.

Answer C: RAID 5

The changing parity disk means parity is distributed across the array while data are striped in blocks, identifying RAID5. Its usable capacity is (N−1)D, so four 2 TB disks provide 6 TB and retain one disk's worth of redundancy. RAID2 uses bit striping with Hamming-code redundancy, RAID1 mirrors data, and RAID0 provides no disk-failure tolerance.

Why each choice is right or wrong

A - Incorrect: RAID2 uses bit striping with Hamming-code redundancy, not the described block stripes with rotating parity.

B - Incorrect: RAID1 mirrors data; it does not place one rotating parity block in each stripe.

C - Correct: RAID5 distributes parity across block stripes and gives (4−1)×2 = 6 TB usable capacity.

D - Incorrect: RAID0 would provide 8 TB without parity, so it fails the one-disk-failure requirement.

Primary note proof

Study-note section NSA25

> RAID0 stripes for capacity/throughput with no disk-failure tolerance. RAID1 mirrors; a two-disk mirror has one disk's usable capacity and survives one disk failure. RAID2 uses bit striping/Hamming-code redundancy, RAID3 byte striping with dedicated parity, RAID4 block striping with dedicated parity and RAID5 block striping with distributed parity. RAID6 distributes dual parity and tolerates two disk failures. RAID10 stripes mirrored pairs; survival depends on which disks fail.

Study-note section NSA25

> With N equal disks of capacity D, RAID0 usable=ND, paired RAID1/10≈ND/2, RAID5=(N−1)D and RAID6=(N−2)D. Hardware RAID may offload work to controller resources/cache but does not universally outperform software RAID. Rebuilds consume bandwidth; parity writes have

overhead. A large university/bank alone does not select a level. Striping plus parity is not uniquely RAID5 unless distributed block parity is specified.

Question sources

Computer Science Exit Exam · 2018 · Question 50

Question 101 | NS0350

Which one of the following is WRONGLY stated about an importance of security policy?

A. To help minimize risk.

B. To ensure the confidentiality, integrity and availability of data.

C. To coordinate and enforce a security program across an organization.

D. To use the system in a passive way

Hint

Distinguish the security objective from the control that prevents, detects or recovers from the stated problem.

Answer D: Passive system use is not a security-policy purpose

NSA26 assigns policies risk reduction, coordinated required behavior and CIA goals. D's passive-use statement does not describe those governance/protection purposes and is explicitly rejected here. B expresses the desired CIA objective, not a guarantee that policy alone prevents every incident. The remaining statements fit legitimate policy aims.

Why each choice is right or wrong

A - Incorrect: Reducing risk is a stated security-policy aim, so this is not wrongly stated.

B - Incorrect: CIA protection is a legitimate policy objective, with actual success depending on implemented controls.

C - Incorrect: Coordinating required behavior and enforcement across the organization is a legitimate policy purpose.

D - Correct: Passive use is not the defined purpose of security policy and does not express its governance/protection objectives.

Primary note proof

Study-note section NSA26

A security policy states required behavior, scope, accountable roles and enforcement. Begin with assets/service objectives, assess threats and weaknesses, choose controls, define exceptions/incident procedures, approve/communicate, monitor and periodically review. CIA means confidentiality, integrity and availability; neither sensitive-data protection nor availability alone is the universal primary goal.

Study-note section NSA26

A policy can require password controls, approved encryption, restricted administrative access, backups and patch deadlines; firewall rules implement part of it. Open unrestricted wireless access conflicts with controlled internal access unless intentionally isolated as a guest service. “Use the system passively” is not a purpose of security policy. Policies coordinate action and reduce risk; they do not guarantee every attack will fail.

Question sources

Computer Science Exit Exam · 2018 · Question 85

Automata and Complexity Theory

Questions & Worked Answers

156 questions

Question 1 | Q00015

Which of the following term refers to the number of symbols composing a string?

A. Length

B. Symbols

C. Alphabet

D. Word

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer A: Length

Count symbols rather than distinct symbols: that count is the string's length, $|w|$. Alphabet is the set from which symbols are drawn, not the count in a particular string.

Why each choice is right or wrong

A - Correct: This is precisely the defined count $|w|$.

B - Incorrect: Symbols are the individual objects being counted.

C - Incorrect: An alphabet is a finite set of symbols.

D - Incorrect: A word/string is the sequence whose length is measured.

Primary note proof

Study-note section AUT01

> An alphabet Σ is a finite set of symbols. A word/string is a finite ordered sequence of symbols from Σ; a language is a set of words, $L \subseteq \Sigma^*$. An empty word ε, also written λ, has length zero. The empty language $\varnothing$ contains no words; $\{\varepsilon\}$ contains one.

Study-note section AUT01

> Length counts occurrences, not distinct symbols: $|caddba|=6$, so $|caddba|^2=36$.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 2 | lines 26-26

> The number of symbols in a string w is called the length of a string. It is denoted by $|w|$.

Automata and Complexity Theory module | PDF page 2 | lines 15-15

> An alphabet finite set of non-empty symbols. It is denoted by ∑ (sigma).

Automata and Complexity Theory module | PDF page 2 | lines 21-21

> It is a finite collection of symbols from the alphabet. The string is denoted by w.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 15

Question 2 | Q00024

Let L_1={c,d} and L_2={e,fgh}. Which lowercase word is NOT an element of L_2L_1?

A. fghc

B. ed

C. fghe

D. ec

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer C: fghe

Language concatenation preserves order: each word of L_2 is followed by a word of L_1. The four products are ec, ed, fghc and fghd. Thus fghe is the unique nonmember: its ending e is not one of the words available from L_1. The lowercase transcription repair makes the symbols consistent with the declared languages.

Why each choice is right or wrong

A - Incorrect: fghc is fgh followed by c, so it belongs to the product.

B - Incorrect: ed is e followed by d, so it belongs to the product.

C - Correct: fghe cannot be formed because e is not in L_1.

D - Incorrect: ec is e followed by c, so it belongs to the product.

Primary note proof

Study-note section AUT02

> Concatenation xy writes x followed by y: cddc·ababb=cddcababb; order generally matters. εx=xε=x. Define x^0=ε and x^n as n copies for n≥1. Reverse x^R reads symbols backward: caddba reverses to abddac; $(xy)^R=y^Rx^R$ and $(x^R)^R=x$.

Study-note section AUT02

For languages, LM={xy:x∈L,y∈M}, L^0={ε}, and L^{n+1}=L^nL. Thus {e,fgh}{c,d}={ec,ed,fghc,fghd}; fghe is absent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 2 | lines 21-21

It is a finite collection of symbols from the alphabet. The string is denoted by w.

Automata and Complexity Theory module | PDF page 3 | lines 13-14

A grammar defines a set of rules, and with the use of these rules, valid sentences in a language are
generated.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 24

Question 3 | Q00032

Which theory provides mechanisms for classifying combinatorial problems and measuring the computational resources necessary to solve them?

A. Computability theory

B. Set theory

C. Complexity theory

D. Automata theory

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer C: Complexity theory

The requested quantity is the resources needed to solve a problem. Computational complexity estimates time and space requirements. Computability instead asks whether a membership algorithm exists at all.

Why each choice is right or wrong

A - Incorrect: Computability concerns existence of algorithms, rather than their resource costs.

B - Incorrect: Set theory does not name the study of time and space requirements.

C - Correct: Matches the resource-estimation definition directly.

D - Incorrect: Automata studies abstract machines; the specific resource classification asked here is complexity.

Primary note proof

Study-note section AUT-S01

> Computability theory asks whether an effective algorithm can solve a problem at all. For a decision problem, a decision procedure must terminate with the correct yes-or-no result on every input.

Study-note section AUT-S01

> Computational complexity theory studies the time and memory resources required to solve problems as functions of encoded input size. Automata theory studies abstract machine models and the languages they recognize; these subjects overlap but ask different questions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 15 | lines 33-35

> Algorithm analysis is an important part of a broader computational complexity theory, which provides theoretical estimates for the resources needed by any algorithm which solves a given computational problem.

Automata and Complexity Theory module | PDF page 15 | lines 24-26

> A set X is recursive if we have an algorithm to determine whether a given element belongs to X or not.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 32

Question 4 | Q00192

Which language over {0,1} is denoted by (0+1)*0(0+1)*0(0+1)*? Here + means union.

A. Strings with substring 00

B. Strings with at least two 0's

C. Strings with at most two 0's

D. Strings with postfix 1.

Hint

Track mandatory symbols and factors that may repeat or disappear. Use short test words to check both directions of each proposed language description.

Answer B: Strings with at least two 0's

Each starred factor denotes any binary string, including the empty string. The two unstarred zeros therefore require at least two zero occurrences, while permitting arbitrary symbols before, between and after them. Conversely, any word with at least two zeros can be split at two selected zero positions to fit the expression. For example, 010 fits without containing adjacent zeros, and 000 fits although it has more than two zeros. No final 1 is required.

Why each choice is right or wrong

A - Incorrect: 010 is generated but does not contain the adjacent substring 00.

B - Correct: Two explicit zeros are required; the starred factors allow additional zeros and ones.

C - Incorrect: 000 is generated, so there is no upper bound of two zeros.

D - Incorrect: 00 is generated and ends in 0.

Primary note proof

Study-note section AUT13

> To require at least one a anywhere over {a,b}, use (a|b)*a(a|b)*. b*ab* requires exactly one a. At least two zeros uses (0|1)*0(0|1)*0(0|1)*; the zeros need not be adjacent. Testing 010 distinguishes it from a requirement for substring 00. For S→aS|ε, any number of a's is generated, giving a*.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

> A language is said to be a Regular language if and only if some finite state machine recognizes it

Question sources

Addis Ababa University mock exam · Question 94

Question 5 | Q00193

The transition function in a finite-state automaton maps which pair?

A. Q and Σ to Σ

B. Q and Q to Σ

C. Σ and Σ to Q

D. Q and Σ to Q

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer D: Q and Σ to Q

For a deterministic finite automaton, finite control receives the current state and input symbol and chooses the next state. Thus $\delta:Q\times\Sigma\to Q$. A general NFA instead has a set of possible next states, so the stem tacitly uses the deterministic convention.

Why each choice is right or wrong

A - Incorrect: Outputs an input symbol, not a state.

B - Incorrect: The second input must be a symbol, not another state.

C - Incorrect: Omits the current state and uses two input symbols.

D - Correct: Correct deterministic domain and codomain.

Primary note proof

Study-note section AUT06

> A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0\in Q$, accepting set $F\subseteq Q$ and total $\delta:Q\times\Sigma\to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition. F may be empty, recognizing $\varnothing$.

Study-note section AUT06

> Start at q_0 and consume the entire input. Accept iff the ending state is in F; visiting an accepting state earlier is insufficient. ε is accepted exactly when $q_0\in F$. A nonaccepting sink loops on every symbol and cannot reach acceptance; it still finishes after finite input. Finite automata have finite memory encoded by states, not no state or no ability to remember any history.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 3 | lines 25-25

> A finite automaton is a collection of 5-tuple (Q, ∑, δ, q0, F), where:

Automata and Complexity Theory module | PDF page 4 | lines 5-6

> Finite control: The finite control decides the next state on receiving particular input from input tape.

Automata and Complexity Theory module | PDF page 5 | lines 3-5

> DFA refers to deterministic finite automata. Deterministic refers to the uniqueness of the computation. In the DFA, the machine goes to one state only for a particular input character. DFA does not accept the null move.

Question sources

Addis Ababa University mock exam · Question 95

Question 6 | Q00194

With start variable A and productions A→aB|bA|bBa and B→bB|cA|b, which offered string is generated?

A. abbcabbcbbbcbbaca

B. bcbbbcabbcbaa

C. acbcbbbbcabaa

D. bbbbcbbbbcab

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer C: acbcbbbbcabaa

With A as the start variable, the candidate follows the production sequence A⇒aB⇒acA⇒acbBa⇒acbcAa⇒acbcbAa⇒acbcbbAa⇒acbcbbbAa⇒acbcbbbbBaa⇒acbcbbbbcAaa⇒acbcbbbbcaBaa⇒acbcbbbbcabaa. Each rewrite uses one of the displayed rules, including a final B→b that removes the remaining variable. The other three offered words fail the complete shrinking-residual membership test.

Why each choice is right or wrong

A - Incorrect: Every permitted prefix or wrapper decomposition eventually leaves a residual that neither required variable generates.

B - Incorrect: The required A/B prefix and wrapper decompositions do not reach the terminating B→b production.

C - Correct: The displayed derivation produces acbcbbbbcabaa from start A.

D - Incorrect: Following the required variable after each prefix does not produce this word.

Primary note proof

Study-note section AUT40

> For A→aB|bA|bBa and B→bB|cA|b, specify a start variable first. With A assumed, a terminating membership test recursively removes the required a/b prefix or b…a wrapper; B either matches b or removes b before B/c before A. Every recursive residual is shorter. Apply the membership test to every candidate word; none qualifies in this four-word example under the assumed start A. Use the membership test to justify whether a candidate word belongs to the language.

Study-note section AUT16

> In a context-free grammar (CFG) each production has one variable on the left, A→α with α∈(V∪Σ)*.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string $s \in (V \cup \Sigma)*$;

Automata and Complexity Theory module | PDF page 9 | lines 2-2

> A sentential form is any string derivable from the start symbol.

Question sources

Addis Ababa University mock exam · Question 96

Question 7 | Q00195

When can we say two grammars are equivalent?

A. When the language derived from one grammar is a subset of the other

B. When the alphabets of the two grammars are the same

C. When the languages represented by both grammars are the same

D. When a single string is a member of both languages

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer C: When the languages represented by both grammars are the same

A grammar specifies its generated sentences. Equality of the generated sets, not equality of alphabets or overlap, is the criterion for semantic equivalence. For example {a} and {a,aa} have the same alphabet and a common string but different languages.

Why each choice is right or wrong

A - Incorrect: One-way inclusion allows extra strings in the other language.

B - Incorrect: Same terminal alphabet permits different sets of sentences.

C - Correct: Equality of the full generated languages establishes equivalence.

D - Incorrect: A single shared string establishes overlap only.

Primary note proof

Study-note section AUT04

> Write $G=(V,\Sigma,P,S)$: finite variables V, terminals Σ disjoint from V, productions P and start variable $S \in V$. A production replaces its left side by its right side. The language is $L(G)=\{w \in \Sigma^* : S \Rightarrow^* w\}$. A recognizer processes a word and accepts exactly the language members. Grammars generate languages; automata recognize languages, not grammars.

Study-note section AUT04

> Two grammars are equivalent when their generated languages are equal, regardless of rule counts or variable names. One common word, equal alphabets or one-way inclusion is insufficient.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 3 | lines 13-14

> A grammar defines a set of rules, and with the use of these rules, valid sentences in a language are
> generated.

Question sources

Addis Ababa University mock exam · Question 97

Computer Science Exit Exam · Sene 2016 · Question 19-3

Computer Science Exit Exam practice collection · Question 262

Question 8 | Q00196

If L1 and L2 are two regular languages, which one of the following is true

A. LiU L2 is a regular language only if empty string is not a member

B. Li* is the same with L2* and are both regular languages

C. LiL is also a regular language

D. L:* is a regular language and is equivalent to L;

Hint

Track mandatory symbols and factors that may repeat or disappear. Use short test words to check both directions of each proposed language description.

Answer C: The concatenation L1L2 is regular.

Regular languages are closed under concatenation, so any two regular languages have a regular concatenation. Reversal preserves regularity but need not preserve the original language. Union needs no exclusion of the empty word, and arbitrary closures need not coincide.

Why each choice is right or wrong

A - Incorrect: Union is regular even if it contains the empty word; the only-if restriction is false.

B - Incorrect: Both stars are regular, but arbitrary regular languages need not have equal Kleene closures.

C - Correct: The image reads L1L2; regular languages are closed under this concatenation operation.

D - Incorrect: The image reads reversal: it remains regular but need not equal L1, for example {ab} reverses to {ba}.

Primary note proof

Study-note section AUT14

> Right-linear rules have form A→aB, A→a or permitted ε rules; left-linear rules put the single variable at the opposite end, A→Ba. Both consistent orientations generate REG; arbitrarily mixing them can generate nonregular languages, for example S→aA|ε, A→Sb generates a^nb^n. A CFG rule A→aBA is legal but not regular-grammar form.

Study-note section AUT14

> Regular languages are closed under union, concatenation, star, reversal, intersection, difference and complement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

> A language is said to be a Regular language if and only if some finite state machine recognizes it

Automata and Complexity Theory module | PDF page 6 | lines 4-4

> Kleen closure: If L is a regular language then its Kleen closure L1* will also be a regular language.

Question sources

Addis Ababa University mock exam · Question 98

Question 9 | Q00197

Let G = {N, T, P, S} be a grammar where:
N={S, A}, T={ 0, 1}, S={S} and P= {S-11S|0A]|1A,A>0]|1] 0S}.
Which one of the following is a sentential form?

A. 111011110S

B. 11111101

C. 0000111001AS

D. OOO011L1I0A

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer B: 11111101

The grammar is S→11S|0A|1A, A→0|1|0S, and option D is 00001110A. Terminal sentences are also sentential forms. B has the derivation S⇒11S⇒1111S⇒111111S⇒1111110A⇒11111101. Every reachable form contains at most one nonterminal, at the right end. Before final termination, a return to S appends either 11, 00, or 10. A cannot be segmented into these pairs (its last prefix pair is 01); C has two variables; D has prefix ending at A which cannot be obtained by those pairs followed by 0 or 1.

Why each choice is right or wrong

A - Incorrect: Before S the terminal prefix must be a sequence of pairs 11,00,10; 111011110 contains a forbidden pair 01 and has odd length.

B - Correct: The explicit derivation ends at this terminal sentence, which is also a sentential form.

C - Incorrect: Two variables AS cannot arise from rules that never introduce two variables.

D - Incorrect: The form 00001110A would require an odd-length prefix before A (pairs followed by one symbol); its eight-symbol prefix has even length.

Primary note proof

Study-note section AUT40

> For S→11S|0A|1A and A→0|1|0S, every reachable form has at most one variable, at the right end. A complete terminal word is still a sentential form. For example S⇒11S⇒1111S⇒111111S⇒1111110A⇒11111101. Returning to S appends a pair 11,00 or 10; use that invariant to rule out impossible prefixes rather than guessing from string length.

Study-note section AUT16

> A sentential form is any string derived from S, including terminal-only sentences and S itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 9 | lines 2-2

> A sentential form is any string derivable from the start symbol.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string $s \in (V \cup \Sigma)*$;

Question sources

Addis Ababa University mock exam · Question 99

Question 10 | Q00198

In a derivation graph whose root is S, whose vertices are sentential forms, and whose directed edges are single production applications, which property records the steps generating a terminal string?

A. Root of the graph

B. Degree of the vertex

C. Path of a vertex from the root

D. Parent vertex

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer C: Path of a vertex from the root

A derivation is a sequence of production applications beginning at S and ending at a terminal word. Under the specified graph model, each application is one directed edge, so the root-to-terminal-vertex path records the complete sequence. A root alone gives only the start; degree and an immediate parent give local information. This graph is not being treated as a parse tree.

Why each choice is right or wrong

A - Incorrect: The root identifies S but contains no sequence of applications.

B - Incorrect: Degree counts incident edges rather than their ordered traversal.

C - Correct: A root-to-vertex path lists the successive production applications.

D - Incorrect: A parent identifies one predecessor rather than the entire derivation.

Primary note proof

Study-note section AUT16

> In a context-free grammar (CFG) each production has one variable on the left, $A \to \alpha$ with $\alpha \in (V \cup \Sigma)^*$. The left side therefore also belongs to $V \cup \Sigma$ and its string closure; alternatives stating these weaker memberships can all be true. A sentential form is any string derived from S, including terminal-only sentences and S itself.

Study-note section AUT16

> A leftmost derivation always replaces the leftmost variable; a rightmost one replaces the rightmost variable. This is not scanning the completed input in opposite directions. A complete parse tree has root S, internal variable nodes expanded by productions and terminal leaves, with ε leaves when that convention is used. Read the leaves left-to-right for the yielded word. The whole tree represents derivation structure; a path in a derivation graph is a different object.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 30-31

> The nodes of the tree represent the
> symbols and the edges represent the use of production rules.

Automata and Complexity Theory module | PDF page 9 | lines 5-6

> The derivation tree is a graphical representation of the given production rules of context-free grammar (CFG).

Question sources

Addis Ababa University mock exam · Question 100

Question 11 | Q00329

A context free Grammar is called ambiguous if there exists a string that can have?

A. only one parse tree

B. no parse tree

C. Partial parse tree

D. more than one parse tree

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer D: more than one parse tree

Ambiguity is witnessed by one terminal string having two distinct parse trees. It does not require every string to be ambiguous, and a string with no parse tree is simply not generated.

Why each choice is right or wrong

A - Incorrect: A unique tree is consistent with unambiguity.

B - Incorrect: No tree means failure to derive the string, not competing derivations.

C - Incorrect: A partial tree is unfinished and does not witness two full derivations.

D - Correct: Two or more full trees for the same string meet the definition.

Primary note proof

Study-note section AUT18

> A CFG is ambiguous when some word has two distinct parse trees, equivalently two different leftmost derivations. Different expansion orders that give the same tree do not prove ambiguity.

Study-note section AUT18

> Separate precedence levels: E→E+T|T, T→T*F|F, F→id|(E). Multiplication binds inside T before addition at E, and left recursion encodes left association. Uniqueness follows by the top-level last

operator split and induction. S→SS|ε is already ambiguous for ε. S→A|B is ambiguous if A→ab and B→abB|ε both produce ab. Ambiguous grammar and inherently ambiguous language differ: another grammar may remove ambiguity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 33-34

Grammar can be implemented with multiple parse
trees to get the same resulting string, this is said to be ambiguous.

Question sources

Computer Science model exam · Question 60

University of Gondar model exam · Question 60

Question 12 | Q00330

A context free grammar with production rules: S->aS / ¢ is equivalent to the regular expression

A. a*

B. (atb)*

C. a(at+b)*

D. a+

Hint

Track mandatory symbols and factors that may repeat or disappear. Use short test words to check both directions of each proposed language description.

Answer A: a*

The grammar is S→aS|ε. Applying the recursive rule n times followed by ε produces a^n for each n≥0, and no production creates any other terminal. This is exactly a*, including ε. Positive closure a+ omits ε.

Why each choice is right or wrong

A - Correct: Contains exactly all finite repetitions of a, including zero.

B - Incorrect: The expression (a+b)* also contains b.

C - Incorrect: The expression a(a+b)* contains ab and omits ε.

D - Incorrect: Requires at least one a, although S⇒ε is permitted.

Primary note proof

Study-note section AUT13

To require at least one a anywhere over {a,b}, use (a|b)*a(a|b)*. b*ab* requires exactly one a. At least two zeros uses (0|1)*0(0|1)*0(0|1)*; the zeros need not be adjacent. Testing 010 distinguishes it from a requirement for substring 00. For S→aS|ε, any number of a's is generated, giving a*.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

R is a set of production rules where each production rule maps a variable to a string s∈(V∪Σ)∗;

Automata and Complexity Theory module | PDF page 2 | lines 24-25

A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Question sources

Computer Science model exam · Question 61

University of Gondar model exam · Question 61

Question 13 | Q00331

A push down automaton employs data structure.

A. Linked List

B. Stack

C. Hash Table

D. Queue

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: Stack

A PDA augments finite control with a stack. Push and pop operations supply its unbounded LIFO memory.

Why each choice is right or wrong

A - Incorrect: A linked list is an implementation possibility, not the abstract PDA storage discipline.

B - Correct: A PDA uses stack memory.

C - Incorrect: Hash lookup is not the PDA memory mechanism.

D - Incorrect: A queue is FIFO, unlike the required pushdown stack.

Primary note proof

Study-note section AUT21

> A nondeterministic pushdown automaton (PDA) adds a stack to finite control: $(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$. Its transition relation maps (state,input symbol or ε,stack top) to possible (new state,replacement stack string) pairs.

Study-note section AUT-S02

> A stack is last-in, first-out: push adds an item at the top and pop removes the top item, so the most recently pushed item is removed first. A PDA accesses this top rather than an arbitrary interior position.

Study-note section AUT-S02

> A queue is first-in, first-out: insertion at the rear and removal at the front returns the oldest waiting item. A linked list can implement a storage discipline, but the implementation container is different from the abstract push/pop or enqueue/dequeue access rule.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 9 | lines 18-19

> Pushdown automata are nondeterministic finite state machines augmented with additional memory
> in the form of a stack,

Question sources

Computer Science model exam · Question 62

University of Gondar model exam · Question 62

Question 14 | Q00332

Which of the following Machine is Equivalent for Regular grammar?

A. Push down automata

B. Linear bounded automata

C. Finite state automata

D. Turing Machine

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: Finite state automata

Regular grammars generate exactly the regular languages, which finite-state automata recognize. The visual hierarchy pairs the innermost regular class with finite-state automata. More powerful models can accept those languages too but are not equivalent in class-wide expressive power.

Why each choice is right or wrong

A - Incorrect: PDA languages include nonregular CFLs, so the whole model is more powerful.

B - Incorrect: The diagram pairs LBAs with the larger context-sensitive class.

C - Correct: Matches exactly the class generated by regular grammars.

D - Incorrect: The outer Turing machine class is larger.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 2-2

> The language defined by regular grammar is known as regular language.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

> A language is said to be a Regular language if and only if some finite state machine recognizes it

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Question sources

Computer Science model exam · Question 63

University of Gondar model exam · Question 63

Question 15 | Q00333

For a CFG G=(V,Σ,P,S), consider a production α→β. Treat a symbol also as its length-one string. Which statements hold? Select all that apply.

A. α∈(V∪Σ)

B. α∈V

C. α∈(V∪Σ)*

D. β∈(V∪Σ)*

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer A, B, C, D: α∈(V∪Σ); α∈V; α∈(V∪Σ)*; β∈(V∪Σ)*

A context-free production has exactly one variable on its left and a finite string of variables and terminals on its right. Therefore α∈V. Since V⊆V∪Σ, that same α belongs to the union and, as a one-symbol string, to its Kleene closure. The right-hand side β belongs to (V∪Σ)*, including the possibility of ε. All four memberships hold.

Why each choice is right or wrong

A - Correct: A variable is also an element of V∪Σ.

B - Correct: The left side of a CFG rule is one variable.

C - Correct: That variable is a length-one string in (V∪Σ)*.

D - Correct: The right side is a finite string over variables and terminals.

Primary note proof

Study-note section AUT16

> In a context-free grammar (CFG) each production has one variable on the left, A→α with α∈(V∪Σ)*. The left side therefore also belongs to V∪Σ and its string closure; alternatives stating these weaker memberships can all be true. A sentential form is any string derived from S, including terminal-only sentences and S itself.

Study-note section AUT16

> A leftmost derivation always replaces the leftmost variable; a rightmost one replaces the rightmost variable. This is not scanning the completed input in opposite directions. A complete parse tree has root S, internal variable nodes expanded by productions and terminal leaves, with ε leaves when that convention is used. Read the leaves left-to-right for the yielded word. The whole tree represents derivation structure; a path in a derivation graph is a different object.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

R is a set of production rules where each production rule maps a variable to a string $s \in (V \cup \Sigma)*$;

Automata and Complexity Theory module | PDF page 8 | lines 22-23

Context-free grammars have a start symbol and set of terminals, non-terminals, and production rule components.

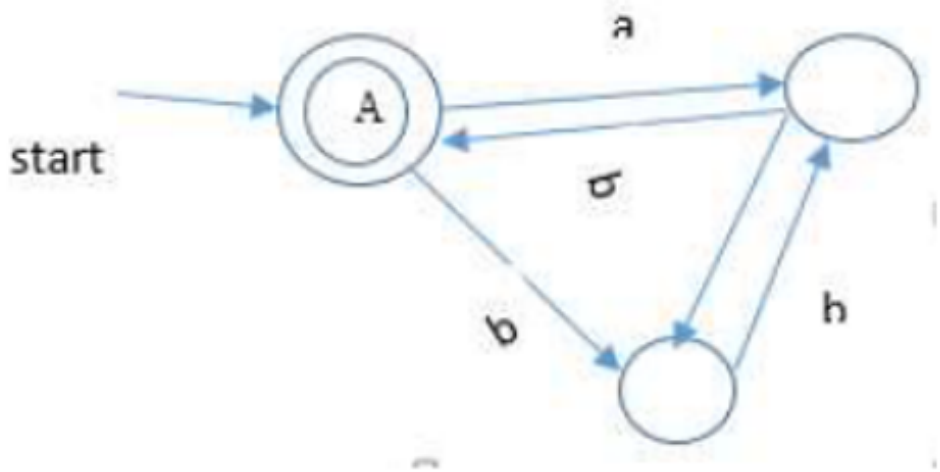
Question sources

Computer Science model exam · Question 64

University of Gondar model exam · Question 64

Question 16 | Q00334

For the diagram, name the top-right state B and bottom state C. Use A as start and sole accepting state; transitions are A-a→B, A-b→C, B-a→C, B-b→A, C-b→B. Unshown transitions reject. Which regular expression denotes its language? Here + means union.



A. a*(a+b)ab(ab+bb+aa*(a+b)ab)*

B. ((a+bb)(ab)*b)*

C. (a+b)ab(ab+bb+aa*(a+b)ab)*

D. a*ab(ab+bb+aa*(a+b)ab)*

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer B: ((a+bb)(ab)*b)*

A return to A first reaches B either directly by a or via C by bb. While at B, the machine may make any number of ab detours through C. It returns to A by b. Thus one return has form (a+bb)(ab)*b, and any number of returns gives ((a+bb)(ab)*b)*. This also accepts ε and ab, as the accepting start and shortest return require.

Why each choice is right or wrong

A - Incorrect: Its mandatory prefix excludes ε, which must be accepted.

B - Correct: Its starred blocks describe exactly all excursions from A back to A.

C - Incorrect: Its mandatory prefix excludes the empty word.

D - Incorrect: Its mandatory ab suffix excludes ε.

Primary note proof

Study-note section AUT12

> Classical expressions use ∅, ε, symbols, union, concatenation and star. A union B is written A|B or A+B; postfix r^+ means one-or-more r repetitions and is a different use of +.

Study-note section AUT13

> For automaton→RE, eliminate a state k while updating edge i→j as $r_{ij} \mid r_{ik}(r_{kk})^*r_{kj}$; include new start/final nodes where required.

Study-note section AUT06

> Start at q_0 and consume the entire input. Accept iff the ending state is in F; visiting an accepting state earlier is insufficient. ε is accepted exactly when $q_0 \in F$.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 3 | lines 25-25

> A finite automaton is a collection of 5-tuple (Q, ∑, δ, q0, F), where:

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

> A language is said to be a Regular language if and only if some finite state machine recognizes it

Question sources

Computer Science model exam · Question 65

University of Gondar model exam · Question 65

Question 17 | Q00634

Which one of the following regular expressions gives zero or one?

A. r + =rr*

B. r* = r+ | ε

C. r? =r|ε

D. All

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer C: The optional expression r? = r|ε.

AUT12 defines r? as r|ε: one occurrence of a word matching r or no occurrence. Positive closure requires one or more occurrences; star permits arbitrarily many including zero. Their identities may be valid, but only the optional expression denotes the requested zero-or-one repetition count, so the aggregate is false.

Why each choice is right or wrong

A - Incorrect: Positive closure rr* requires at least one repetition and also allows more than one.

B - Incorrect: Star allows zero or arbitrarily many repetitions, not just zero or one.

C - Correct: The union with ε makes r optional: either one r match or none.

D - Incorrect: The positive and Kleene closures do not impose the zero-or-one bound.

Primary note proof

Study-note section AUT12

> Classical expressions use ∅, ε, symbols, union, concatenation and star. A union B is written A|B or A+B; postfix r^+ means one-or-more r repetitions and is a different use of +. Parenthesize when conventions might clash. Star binds tightest, concatenation next, union last. r? means r|ε; r^+=rr*.

Study-note section AUT12

> (a|b)*a denotes words ending a; 0*10* denotes exactly one 1; a*(b|c) gives zero-or-more a's then exactly one b or c, so aab belongs but aaa and aaabc do not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 5 | lines 25-25

> means one or more occurrence of x. It can generate {x, xx, xxx, xxxx,

Question sources

Computer Science model exam · 2015 · Question 103

Question 18 | Q00751

If a grammar produces more than one leftmost derivation for a sentence, it is called as:

A. Ambiguous grammar

B. Perfect grammar

C. Associative grammar

D. Precedence

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Ambiguous grammar

Each parse tree determines a unique leftmost sequence of expansions. Two distinct leftmost derivations of the same sentence therefore encode distinct parse trees and witness ambiguity under this definition.

Why each choice is right or wrong

A - Correct: Two leftmost derivations of one sentence witness two parse trees.

B - Incorrect: Perfect is not the relevant defined grammar property.

C - Incorrect: Associativity is an operator grouping convention, not this definition.

D - Incorrect: Precedence concerns ordering of operators; it does not name multiple leftmost derivations.

Primary note proof

Study-note section AUT18

> A CFG is ambiguous when some word has two distinct parse trees, equivalently two different leftmost derivations. Different expansion orders that give the same tree do not prove ambiguity.

Study-note section AUT18

> Separate precedence levels: E→E+T|T, T→T*F|F, F→id|(E). Multiplication binds inside T before addition at E, and left recursion encodes left association. Uniqueness follows by the top-level last operator split and induction. S→SS|ε is already ambiguous for ε. S→A|B is ambiguous if A→ab and B→abB|ε both produce ab. Ambiguous grammar and inherently ambiguous language differ: another grammar may remove ambiguity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 33-34

> Grammar can be implemented with multiple parse
> trees to get the same resulting string, this is said to be ambiguous.

Automata and Complexity Theory module | PDF page 9 | lines 7-8

> in the leftmost derivation, the input is
> scanned and replaced with the production rule from left to right.

Question sources

Computer Science model exam 2 · Question 92

Question 19 | Q00752

The intersection of a context-free language and a regular language is _____

A. context-free

B. neither context-free nor regular.

C. regular but not context-free

D. both regular and context-free.

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Always context-free, but not necessarily regular.

AUT23 constructs the intersection by combining PDA control with DFA control and retaining the original stack. It therefore guarantees a CFL. Regularity does not follow: intersecting $\{a^n b^n\}$ with a*b* gives the same nonregular CFL. Some individual intersections can be regular, but the question's general result cannot assert that extra property.

Why each choice is right or wrong

A - Correct: Closure under intersection with a regular language guarantees that the result is context-free.

B - Incorrect: The stated closure theorem directly rules out a result outside the context-free family.

C - Incorrect: Every regular language is context-free, so regular but not context-free is impossible.

D - Incorrect: Some intersections are regular, but the counterexample in AUT23 shows regularity is not guaranteed.

Primary note proof

Study-note section AUT23

CFLs are closed under union, concatenation, star, reversal and intersection with a regular language. For CFL∩REG combine PDA control with DFA control, using the original stack; each consumed symbol updates both states, while an ε move leaves the DFA state unchanged. The result is guaranteed context-free, not necessarily regular: intersect a^nb^n with a*b* to get a^nb^n.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 9 | lines 23-23

Pushdown automata accept context-free languages, which include the set of regular languages.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

A language is said to be a Regular language if and only if some finite state machine recognizes it

Question sources

Computer Science model exam 2 · Question 93

Question 20 | Q00753

Which one of the following is not part of Turing machine?

A. Finite control

B. Input

C. Tape

D. Head

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: Input is data, not a separate hardware component.

AUT27 lists finite control, a read/write head and an unbounded tape as the standard TM architecture. Input is initial tape content rather than a separate hardware organ. Thus input is the distinction requested by not part, under that structural reading; the other three options are actual machine components, not merely values processed by them.

Why each choice is right or wrong

A - Incorrect: Finite control is explicitly part of the standard machine's architecture.

B - Correct: Input is initial tape content rather than a separate machine hardware component.

C - Incorrect: The unbounded tape is the standard TM's storage component.

D - Incorrect: The read/write head is an explicit machine component used for tape access.

Primary note proof

Study-note section AUT27

> For a Turing machine (TM), use $M=(Q,\Sigma,\Gamma,\delta,q_0,q\text{accept},q\text{reject})$, with finite state/tape alphabets, $\Sigma\subseteq\Gamma$ and blank $\square\in\Gamma-\Sigma$. The blank symbol belongs to the tape alphabet Γ, not the input alphabet Σ. The standard machine has finite control, a read/write head and an unbounded tape. Input is initial tape content, not a separate hardware organ; no separate stack is required.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 11 | lines 9-9

> It has a tape head that can read and write symbols and move around on the tape.

Automata and Complexity Theory module | PDF page 11 | lines 20-22

> δ is the transition function, mapping (q, x) onto (q', y, D) where D denotes the direction of movement of R/W head: D = L or R according as the movement is to the left or right.

Question sources

Computer Science model exam 2 · Question 94

Question 21 | Q00891

Which automaton is powerful enough to recognize the language $L = \{a^n b^n c^n \mid n > 1\}$, where n is an integer?

A. DFA

B. Linear Bounded Automaton (LBA)

C. NFA

D. Pushdown Automaton (PDA)

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer B: Linear Bounded Automaton (LBA).

The language contains equal numbers of a, b and c in that order. This family is context-sensitive but not context-free. Restricting n to integers greater than 1 does not remove the unbounded equal-count requirement. An LBA checks the order and repeatedly marks one a, one b and one c, using only the input-sized tape region. B therefore recognizes the stated language; DFA, NFA and single-stack PDA cannot.

Why each choice is right or wrong

A - Incorrect: A DFA recognizes regular languages; this three-equal-count language is not even context-free.

B - Correct: An LBA can mark matching a, b and c symbols while staying within space proportional to the input length.

C - Incorrect: An NFA recognizes exactly the regular languages, so nondeterminism does not give it the memory needed here.

D - Incorrect: A standard single-stack PDA recognizes context-free languages; the stated three-equal-count language is not context-free.

Primary note proof

Study-note section AUT25

> A context-sensitive rule $\alpha A\beta \to \alpha\gamma\beta$ replaces A in context, with γ nonempty. Equivalently, noncontracting grammar presentations preserve/increase total length, with the isolated ε exception. The condition is on the complete sides, not an unrelated comparison $|\beta| \geq |\alpha|$.

Study-note section AUT25

> A linear bounded automaton (LBA) is a TM with O(n) tape cells on length-n input; the standard CSL characterization uses nondeterministic LBAs. For $a^n b^n c^n$ first verify a*b*c* order, repeatedly mark one unmarked a, b and c, and reject missing/excess symbols.

Study-note section AUT25

> Thus every CSL is decidable.

Study-note section AUT24

> If L is context-free, some p ensures each sufficiently long $s \in L$ has $s=uvxyz$ with $|vxy| \leq p$, $|vy| \geq 1$ and $uv^i xy^i z \in L$ for every $i \geq 0$. The two pieces v and y are pumped together; either may be empty, not both. A non-CFL proof must defeat every allowed decomposition.

Study-note section AUT24

> For $s=a^p b^p c^p$, the window vxy cannot contain both an a and a c. Pumping affects at most two blocks and at least one count. Use i=0 or 2: the unaffected third block retains p while another count changes, or symbol order breaks. Hence $a^n b^n c^n$ is not CFL.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 9 | lines 18-19

> Pushdown automata are nondeterministic finite state machines augmented with additional memory
> in the form of a stack,

Question sources

Computer Science Exit Exam · 2018 · Question 4

Question 22 | Q00917

A recognizer reads input once from left to right and stores all its working memory in a fixed finite set of states. Which universal statement correctly explains why its language can also be described by a context-free grammar?

A. Every regular language is context-free.

B. Every context-free language is regular.

C. Every context-sensitive language is regular.

D. Every recursively enumerable language is recursive

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer A: Every regular language is context-free.

A recognizer whose memory is a fixed finite state has regular-language power. Every regular language is context-free, so a context-free grammar can also describe its language. The reverse inclusion fails: equal-count matching such as {0^n1^n} is context-free but not regular. Context-sensitive languages need not be regular either. Finally, the halting language separates recognizable from decidable languages, so not every recursively enumerable language is recursive.

Why each choice is right or wrong

A - Correct: Every regular language has an equivalent context-free grammar.

B - Incorrect: {0^n1^n : n≥0} is context-free but not regular.

C - Incorrect: The context-sensitive family includes nonregular languages, including equal-count matching languages.

D - Incorrect: The halting language is recursively enumerable but undecidable, disproving this converse.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Study-note section AUT29

A decider halts on every input and accepts exactly its language; these languages are recursive/decidable. A recognizer must halt and accept every member, but on a nonmember may halt-reject or loop. Its languages are recursively enumerable (RE), Turing-recognizable or semidecidable. Looping is nonacceptance, not a rejecting halt. An undecidable language may still have a recognizer.

Study-note section AUT29

Every decidable language is RE, and the containment is strict.

Study-note section AUT06

Finite automata have finite memory encoded by states, not no state or no ability to remember any history.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 20-21

A context-free grammar
can describe all regular languages and more, but they cannot describe all possible languages.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 15 | lines 30-30

All recursive set is recursively enumerable.

Question sources

Computer Science Exit Exam · 2018 · Question 30

Question 23 | Q00934

In the CNF-first route for converting a CFG into Greibach normal form, which initial normalization is performed before the later ordering, substitution and left-recursion work?

A. Convert the grammar into CNF

B. If any production rule in the grammar is not in GNF form, convert it.

C. .
Convert the grammar into GNF

D. If the grammar exists left recursion, eliminate it.

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Convert the grammar into CNF

The specified route begins by putting the grammar into Chomsky normal form. Later ordering and substitution arrange terminal-leading productions and eliminate left recursion as required for Greibach form. CNF is an intermediate normalization in this particular route, not a universal prerequisite for every GNF construction. The other conversion choices state the goal broadly rather than identify this route's initial normalization.

Why each choice is right or wrong

A - Correct: CNF conversion is the initial normalization in the selected route.

B - Incorrect: Converting arbitrary offending rules does not identify the route's initial normalization.

C - Incorrect: GNF conversion is the overall goal, not this preliminary operation.

D - Incorrect: Left recursion must be handled later in this chosen route.

Primary note proof

Study-note section AUT20

> Some algorithms first convert to CNF; this is a method choice, not a universally mandatory first step.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string $s \in (V \cup \Sigma)*$;

Automata and Complexity Theory module | PDF page 9 | lines 2-2

> A sentential form is any string derivable from the start symbol.

Automata and Complexity Theory module | PDF page 9 | lines 15-16

> Simplification essentially comprises the following steps; Reduction of CFG, Removal of Unit Productions, and Removal of Null Productions.

Question sources

Computer Science Exit Exam · 2018 · Question 47

Question 24 | Q00953

Among the listed grammar families in the Chomsky hierarchy, which has the greatest expressive power?

A. Context-free

B. Context-sensitive

C. Unrestricted (Type 0)

D. Regular

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: Unrestricted (Type 0)

Unrestricted, Type 0 grammars generate the recursively enumerable languages, the family recognized by Turing machines. Context-sensitive, context-free and regular grammars generate successively smaller families. Asking for the most expressive listed family makes this an exact hierarchy comparison. It does not claim that every conceivable set of strings is recursively enumerable or that weaker grammars cannot generate complete languages of their own.

Why each choice is right or wrong

A - Incorrect: Context-free grammars generate a smaller family than RE.

B - Incorrect: Context-sensitive grammars generate a smaller family than RE.

C - Correct: Unrestricted grammars generate exactly RE.

D - Incorrect: Regular grammars generate the least expressive listed family.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 3 | lines 13-14

> A grammar defines a set of rules, and with the use of these rules, valid sentences in a language are
> generated.

Automata and Complexity Theory module | PDF page 8 | lines 20-21

> A context-free grammar
> can describe all regular languages and more, but they cannot describe all possible languages.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Question sources

Computer Science Exit Exam · 2018 · Question 66

Question 25 | Q00977

ou have given that Grammar G1 -({S, A, B}, {a, b}, S, {S - AB, A- a, B-> b}).
Which one is terminal symbol in the give?

A. b

B. S

C. .
S- AB,A--a,B-b

D. S,A,B

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer A: b

The given terminal alphabet is {a,b}, so b is a terminal. S, A and B are variables, and the arrow expressions are productions. These categories match the four-component CFG definition.

Why each choice is right or wrong

A - Correct: b belongs to the given terminal alphabet.

B - Incorrect: S belongs to the variables and is the start symbol.

C - Incorrect: This is a set/list of productions, not one terminal symbol.

D - Incorrect: These are the variables, not the terminals.

Primary note proof

Study-note section AUT04

> Write G=(V,Σ,P,S): finite variables V, terminals Σ disjoint from V, productions P and start variable S∈V. A production replaces its left side by its right side. The language is L(G)={w∈Σ*:S⇒*w}. A recognizer processes a word and accepts exactly the language members. Grammars generate languages; automata recognize languages, not grammars.

Study-note section AUT04

> Two grammars are equivalent when their generated languages are equal, regardless of rule counts or variable names. One common word, equal alphabets or one-way inclusion is insufficient.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 22-23

> Context-free grammars have a start symbol and set of terminals, non-terminals, and production rule components.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string $s \in (V \cup \Sigma)*$;

Question sources

Computer Science Exit Exam · 2018 · Question 90

Question 26 | Q00981

Over $\Sigma=\{a,b\}$, classify $L=\{ww^R \mid w \in \Sigma^*\}$, the unmarked even-length palindrome language, by the most specific listed description.

A. Regular

B. Non-deterministic CFL but not deterministic

C. Deterministic context-free

D. Context-sensitive but not context-free

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: Non-deterministic CFL but not deterministic

An NPDA can push a guessed first half, choose its midpoint nondeterministically, and pop matching symbols while reading the reverse half. This binary unmarked palindrome language is not deterministic context-free. It is context-free and nonregular. With exponent R denoting reversal, ww^R is the stated palindrome language; the forward-copy language ww would instead fall outside CFL.

Why each choice is right or wrong

A - Incorrect: Binary unmarked even palindromes are nonregular.

B - Correct: An NPDA recognizes them, but a DPDA cannot recognize this unmarked binary language.

C - Incorrect: The missing center marker prevents deterministic context-free recognition.

D - Incorrect: The language has a CFG, so the claim that it is not context-free is false.

Primary note proof

Study-note section AUT22

> For even palindromes ww^R over {a,b}, push the first half, nondeterministically guess its midpoint, then pop matching symbols. No center marker makes that language context-free but not deterministic context-free. With a marker wcw^R the switch point is visible and a DPDA suffices. Unary even palindromes are merely even-length words and regular. A PDA cannot compare two arbitrary halves in the same forward order: {ww:w∈{a,b}*} is the copy language, not the palindrome language.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string s∈(V∪Σ)∗;

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Question sources

Computer Science Exit Exam · 2018 · Question 94

Question 27 | Q01185

Which of the following statements is false about classes of language in automaton theory?

A. Context-free language is the subset of context-sensitive language

B. Regular language is the subset of context-sensitive language

C. Recursively enumerable language is the superset of regular language

D. Context-sensitive language is a subset of context-free language

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer D: Context-sensitive language is a subset of context-free language

The inspected diagram nests regular inside context-free inside context-sensitive inside recursively enumerable. Thus A, B and C follow the depicted inclusions, while D reverses the middle inclusion. This uses the traditional hierarchy convention, which suppresses technical empty-word qualifications.

Why each choice is right or wrong

A - Incorrect: Context-free is drawn inside context-sensitive.

B - Incorrect: Regular is inside context-free and hence inside context-sensitive.

C - Incorrect: RE is the outer class surrounding regular.

D - Correct: Reverses the diagram: context-sensitive is not contained in the smaller CFL class.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 8 | lines 20-21

> A context-free grammar
> can describe all regular languages and more, but they cannot describe all possible languages.

Question sources

Computer Science model exam 2 · Question 6

Question 28 | Q01206

Suppose you are asked to construct a DFA for the strings ending with an 'n'
length substring. How many minimal states may the DFA require?

A. n

B. n+1

C. n+2

D. n*2

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer B: n+1 states for a fixed valid suffix word.

AUT08 gives n+1 states for a valid fixed suffix of length n: states remember the longest suffix of the consumed input that is a prefix of the target. Distinguishable prefixes establish the lower bound. The answer presumes the target word uses the declared alphabet; no extra rejecting sink is needed for a suffix matcher.

Why each choice is right or wrong

A - Incorrect: n states omit one of the n+1 necessary prefix-match stages.

B - Correct: The note gives one state for each matched-prefix length from zero through n.

C - Incorrect: An extra rejecting sink is unnecessary because later input may always form the required suffix.

D - Incorrect: Doubling n is not the general suffix-matching state bound; accidental numeric equality does not define it.

Primary note proof

Study-note section AUT08

> For a fixed valid word w of length n, a DFA for strings ending in w needs n+1 states representing the longest consumed suffix that is a prefix of w. Overlapping prefixes determine fallback transitions. Distinct prefixes are distinguishable by suitable continuations, proving the lower bound; do not add an unnecessary rejecting sink.

Study-note section AUT08

> For strings starting with w, a complete DFA ordinarily has n matched-prefix stages, an accepting continuation state and a rejecting sink: n+2 when n≥1 and the alphabet allows mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 3-5

> DFA refers to deterministic finite automata. Deterministic refers to the uniqueness of the computation. In the DFA, the machine goes to one state only for a particular input character. DFA does not accept the null move.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

> A language is said to be a Regular language if and only if some finite state machine recognizes it

Question sources

Computer Science model exam 2 · Question 27

Question 29 | Q01223

Under the standard total-DFA model, which statements about finite automata are NOT correct? Select all that apply.

A. Finite automaton accept a string or word by final state

B. Finite Automaton has at least one final state

C. Finite Automaton has at least one initial state

D. Deterministic Finite Automaton has a single transition.

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer B, D: Finite Automaton has at least one final state; Deterministic Finite Automaton has a single transition.

A DFA requires a designated start state and accepts when the state after the complete input is accepting. Its accepting set may nevertheless be empty, producing the empty language, so B is false. Determinism requires one successor for each state/symbol pair, not one transition in the entire automaton, making D false as well. The two remaining statements describe standard acceptance and initialization.

Why each choice is right or wrong

A - Incorrect: Final-state acceptance after all input is consumed is correct, so A is not selected.

B - Correct: The accepting set can be empty, making the asserted minimum of one false.

C - Incorrect: A designated initial state is required, so C is not selected.

D - Correct: One successor per state/symbol pair does not mean one transition in the entire machine.

Primary note proof

Study-note section AUT06

> A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0 \in Q$, accepting set $F \subseteq Q$ and total $\delta: Q \times \Sigma \to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition. F may be empty, recognizing $\varnothing$.

Study-note section AUT06

> Start at q_0 and consume the entire input. Accept iff the ending state is in F; visiting an accepting state earlier is insufficient. ε is accepted exactly when $q_0 \in F$. A nonaccepting sink loops on every symbol and cannot reach acceptance; it still finishes after finite input. Finite automata have finite memory encoded by states, not no state or no ability to remember any history.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 3 | lines 25-25

A finite automaton is a collection of 5-tuple (Q, ∑, δ, q0, F), where:

Automata and Complexity Theory module | PDF page 5 | lines 3-5

DFA refers to deterministic finite automata. Deterministic refers to the uniqueness of the computation. In the DFA, the machine goes to one state only for a particular input character. DFA does not accept the null move.

Question sources

Computer Science model exam 2 · Question 44

Question 30 | Q01253

Finite automaton recognizes:

A. Any class of language.

B. Context-free language.

C. Context-sensitive languages.

D. Regular languages.

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer D: Regular languages.

Regular languages are exactly those recognized by finite-state automata. This is an exact language-family question: some CFLs and context-sensitive languages happen to be regular, but the whole larger families are beyond finite automata.

Why each choice is right or wrong

A - Incorrect: The hierarchy shows larger families outside finite-state power.

B - Incorrect: All CFLs is too broad, though regular examples are CFLs.

C - Incorrect: All context-sensitive languages is too broad.

D - Correct: Exactly matches the finite-state/regular equivalence.

Primary note proof

Study-note section AUT06

A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0 \in Q$, accepting set $F \subseteq Q$ and total $\delta: Q \times \Sigma \to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition. F may be empty, recognizing ∅.

Study-note section AUT06

Start at q_0 and consume the entire input. Accept iff the ending state is in F; visiting an accepting state earlier is insufficient. ε is accepted exactly when $q_0 \in F$. A nonaccepting sink loops on every

> symbol and cannot reach acceptance; it still finishes after finite input. Finite automata have finite memory encoded by states, not no state or no ability to remember any history.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

> A language is said to be a Regular language if and only if some finite state machine recognizes it

Automata and Complexity Theory module | PDF page 8 | lines 20-21

> A context-free grammar
> can describe all regular languages and more, but they cannot describe all possible languages.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Question sources

Computer Science model exam 2 · Question 74

Question 31 | Q01273

Which one of the following grammars is unambiguous?

A. E → I; E → E + E; E → E * E; E → (E); I → ε | 0 | 1 | 2 | ... | 9

B. S → aSb | SS; S → ε

C. E → E + T / T; T → T x F / F; F → id

D. S → A / B; A → aAb / ab; B → abB / ∈

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: E → E + T / T; T → T x F / F; F → id

In C, F generates id, T is a nonempty sequence of Fs separated by ×, and E a nonempty sequence of Ts separated by +. The last + uniquely determines the top E→E+T split; each T is uniquely split at its last ×. Induction on operator count gives a unique parse. A allows different groupings of id-like operands through E+E and E*E. B already has two trees for ε: S→ε and S→SS→εε. D is also ambiguous: ab is generated by S→A→ab and S→B→abB→ab.

Why each choice is right or wrong

A - Incorrect: For 1+1*1, E→E+E permits a top + parse while E→E*E permits a top * parse; I→1 supplies the operands. These are different trees for the same string (PDF page 8).

B - Incorrect: ε has both a direct leaf expansion and an SS expansion with two empty children.

C - Correct: Operator levels fix the unique split at each stage, giving a unique tree.

D - Incorrect: ab is obtained via either A or B, yielding distinct root expansions.

Primary note proof

Study-note section AUT18

> A CFG is ambiguous when some word has two distinct parse trees, equivalently two different leftmost derivations. Different expansion orders that give the same tree do not prove ambiguity.

Study-note section AUT18

> Separate precedence levels: E→E+T|T, T→T*F|F, F→id|(E). Multiplication binds inside T before addition at E, and left recursion encodes left association. Uniqueness follows by the top-level last operator split and induction. S→SS|ε is already ambiguous for ε. S→A|B is ambiguous if A→ab and B→abB|ε both produce ab. Ambiguous grammar and inherently ambiguous language differ: another grammar may remove ambiguity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 33-34

> Grammar can be implemented with multiple parse
> trees to get the same resulting string, this is said to be ambiguous.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a
> string s∈(V∪Σ)*;

Question sources

Computer Science model exam 2 · Question 94

Question 32 | Q01418

Which language family is recognized exactly by nondeterministic pushdown automata?

A. Type 0 language

B. Type 1 language

C. Type 2 language

D. Type 3 language

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: Type 2 language

Nondeterministic pushdown automata recognize exactly the context-free languages, corresponding to Type 2 grammars. Regular languages are included in that family, but Type 3 is too narrow to describe all PDA languages. Context-sensitive and unrestricted languages include non-context-free languages that an ordinary single-stack PDA cannot recognize. The exact-family wording therefore makes Type 2 the unique classification.

Why each choice is right or wrong

A - Incorrect: Type 0 includes languages beyond PDA recognition.

B - Incorrect: Type 1 includes non-context-free languages beyond PDA recognition.

C - Correct: Type 2 is exactly the context-free family characterized by NPDAs.

D - Incorrect: Type 3 covers only the regular subfamily of PDA languages.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 9 | lines 23-23

> Pushdown automata accept context-free languages, which include the set of regular languages.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Question sources

Compiler Design review questions · Question 26

Question 33 | Q01419

Among context-sensitive, context-free and regular grammar families, which is most expressive under the usual isolated-start ε convention?

A. Context-sensitive grammar

B. Context-free grammar

C. Regular grammar

D. All of these

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Context-sensitive grammar

The hierarchy places regular languages strictly inside context-free languages and context-free languages strictly inside context-sensitive languages. Context-sensitive grammars are therefore the most expressive of the three named families. Unrestricted grammars would be stronger if included, but the question explicitly limits the comparison. The three named classes are not tied for maximum expressive power, so the aggregate alternative does not apply.

Why each choice is right or wrong

A - Correct: Context-sensitive is the largest of the three compared families.

B - Incorrect: Context-free is strictly below context-sensitive.

C - Incorrect: Regular is strictly below both other named families.

D - Incorrect: The three families are not equally most expressive.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 3 | lines 13-14

> A grammar defines a set of rules, and with the use of these rules, valid sentences in a language are
> generated.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

Fig: The traditional Chomsky hierarchy

Question sources

Compiler Design review questions · Question 27

Question 34 | Q01426

In which derivation the right-most non-terminal symbol is replaced at each step?

A. Right look ahead

B. Right claim

C. Rightmost

D. Right non-terminal

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer C: Rightmost

At every derivation step choose the rightmost remaining variable and replace it by a production right side. That is a rightmost derivation.

Why each choice is right or wrong

A - Incorrect: Lookahead is not the name of the expansion order.

B - Incorrect: Right claim does not name a derivation method.

C - Correct: Names the specified rightmost expansion rule.

D - Incorrect: Names a symbol position but not the derivation method.

Primary note proof

Study-note section AUT16

In a context-free grammar (CFG) each production has one variable on the left, $A \to \alpha$ with $\alpha \in (V \cup \Sigma)^*$. The left side therefore also belongs to $V \cup \Sigma$ and its string closure; alternatives stating these weaker memberships can all be true. A sentential form is any string derived from S, including terminal-only sentences and S itself.

Study-note section AUT16

A leftmost derivation always replaces the leftmost variable; a rightmost one replaces the rightmost variable. This is not scanning the completed input in opposite directions. A complete parse tree has root S, internal variable nodes expanded by productions and terminal leaves, with ε leaves when that convention is used. Read the leaves left-to-right for the yielded word. The whole tree represents derivation structure; a path in a derivation graph is a different object.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 9 | lines 9-10

> in the rightmost derivation, the input is scanned and
> replaced with the production rule from right to left.

Question sources

Compiler Design review questions · Question 34

Question 35 | Q01427

In which derivation the leftmost non-terminal symbol is replaced at each step?

A. Left recursion

B. Left non-terminal

C. Left pushdown

D. Leftmost

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer D: Leftmost

Choose the leftmost variable at each step, rather than arbitrarily choosing any remaining variable. This is precisely leftmost derivation.

Why each choice is right or wrong

A - Incorrect: Left recursion is a form of production, not the choice of variable at each derivation step.

B - Incorrect: A left nonterminal is a location, not the derivation name.

C - Incorrect: Left pushdown is not the defined method.

D - Correct: Matches left-to-right variable expansion.

Primary note proof

Study-note section AUT16

> In a context-free grammar (CFG) each production has one variable on the left, A→α with α∈(V∪Σ)*. The left side therefore also belongs to V∪Σ and its string closure; alternatives stating these weaker memberships can all be true. A sentential form is any string derived from S, including terminal-only sentences and S itself.

Study-note section AUT16

A leftmost derivation always replaces the leftmost variable; a rightmost one replaces the rightmost variable. This is not scanning the completed input in opposite directions. A complete parse tree has root S, internal variable nodes expanded by productions and terminal leaves, with ε leaves when that convention is used. Read the leaves left-to-right for the yielded word. The whole tree represents derivation structure; a path in a derivation graph is a different object.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 9 | lines 7-8

in the leftmost derivation, the input is
scanned and replaced with the production rule from left to right.

Question sources

Compiler Design review questions · Question 35

Question 36 | Q01429

Which symbol is not related to context-free grammar?

A. End symbol

B. Start symbol

C. Non-terminal symbol

D. Terminal symbol

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer A: End symbol

A CFG consists of a start symbol, terminals, nonterminals and productions. A dedicated end symbol is not one of the four CFG components, even though a parser may add an input end marker for a separate purpose.

Why each choice is right or wrong

A - Correct: Not a required CFG component in the four-element definition.

B - Incorrect: The distinguished S is required.

C - Incorrect: Variables/nonterminals form V.

D - Incorrect: The alphabet Σ supplies terminals.

Primary note proof

Study-note section AUT04

Write G=(V,Σ,P,S): finite variables V, terminals Σ disjoint from V, productions P and start variable S∈V. A production replaces its left side by its right side. The language is L(G)={w∈Σ*:S⇒*w}. A recognizer processes a word and accepts exactly the language members. Grammars generate languages; automata recognize languages, not grammars.

Study-note section AUT04

Two grammars are equivalent when their generated languages are equal, regardless of rule counts or variable names. One common word, equal alphabets or one-way inclusion is insufficient.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 22-23

Context-free grammars have a start symbol and set of terminals, non-terminals, and production rule components.

Question sources

Compiler Design review questions · Question 37

Question 37 | Q01432

Which of the following tree is the pictorial identification of the derivation?

A. The oct tree

B. The parse tree

C. The binary tree

D. None of the above

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer B: The parse tree

A parse/derivation tree labels nodes with grammar symbols and uses edges to show applications of productions. That is the pictorial representation asked for; it need not be a binary tree because productions may have arbitrary finite right sides.

Why each choice is right or wrong

A - Incorrect: An octree is not the grammar representation defined here.

B - Correct: A parse tree is this graphical representation.

C - Incorrect: Production right sides need not have two children, so binary tree is not the general answer.

D - Incorrect: The parse-tree option already supplies the defined representation.

Primary note proof

Study-note section AUT16

> In a context-free grammar (CFG) each production has one variable on the left, A→α with α∈(V∪Σ)*. The left side therefore also belongs to V∪Σ and its string closure; alternatives stating these weaker memberships can all be true. A sentential form is any string derived from S, including terminal-only sentences and S itself.

Study-note section AUT16

> A leftmost derivation always replaces the leftmost variable; a rightmost one replaces the rightmost variable. This is not scanning the completed input in opposite directions. A complete parse tree has root S, internal variable nodes expanded by productions and terminal leaves, with ε leaves when that convention is used. Read the leaves left-to-right for the yielded word. The whole tree represents derivation structure; a path in a derivation graph is a different object.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 9 | lines 5-6

> The derivation tree is a graphical representation of the given production rules of context-free grammar (CFG).

Automata and Complexity Theory module | PDF page 8 | lines 30-31

> The nodes of the tree represent the
> symbols and the edges represent the use of production rules.

Question sources

Compiler Design review questions · Question 40

Question 38 | Q01435

Which grammar forms are guaranteed to have an equivalent DFA? Select all that apply. Use consistently linear orientations throughout each grammar.

A. Generic grammar (arbitrary unrestricted grammar)

B. Left Linear Grammar

C. Right-linear grammar

D. All of the above

Hint

Track mandatory symbols and factors that may repeat or disappear. Use short test words to check both directions of each proposed language description.

Answer B, C: Left Linear Grammar; Right-linear grammar

Consistently left-linear and consistently right-linear grammars both generate regular languages. Every regular language has an equivalent DFA; a right-linear grammar can first be converted to an NFA and then determinized. Arbitrary unrestricted grammars can generate nonregular languages and have no such guarantee. The all-of-the-above alternative fails because that generic case is not constrained to regular languages.

Why each choice is right or wrong

A - Incorrect: An arbitrary unrestricted grammar need not generate a regular language.

B - Correct: A consistently left-linear grammar generates a regular language.

C - Correct: A consistently right-linear grammar generates a regular language.

D - Incorrect: The generic-grammar case prevents all of the above from being true.

Primary note proof

Study-note section AUT14

> Right-linear rules have form A→aB, A→a or permitted ε rules; left-linear rules put the single variable at the opposite end, A→Ba. Both consistent orientations generate REG; arbitrarily mixing them can generate nonregular languages, for example S→aA|ε, A→Sb generates $a^n b^n$. A CFG rule A→aBA is legal but not regular-grammar form.

Study-note section AUT14

> Regular languages are closed under union, concatenation, star, reversal, intersection, difference and complement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 2-2

> The language defined by regular grammar is known as regular language.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

> A language is said to be a Regular language if and only if some finite state machine recognizes it

Question sources

Compiler Design review questions · Question 43

Question 39 | Q01437

Classify FORTRAN's computational power under an idealized model with ordinary loops and unbounded indexed read/write storage. Which description is intended, rather than a classification of its token patterns or source syntax?

A. Turing-complete language

B. Context-sensitive language

C. Context-free language

D. Regular language

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Turing-complete language

Under the stated idealization, indexed writable storage can represent a Turing tape, with an index recording the head position and variables recording the finite control. A loop implements successive read, write, move and state-update steps. This gives the intended Turing-complete computational classification. The other choices classify word-language families rather than naming this general simulation capability; the conclusion depends on the explicit storage model.

Why each choice is right or wrong

A - Correct: Turing-complete describes the specified general computation capability.

B - Incorrect: Context-sensitive classifies formal word languages rather than this simulation capability.

C - Incorrect: Context-free syntax does not itself characterize general computation power.

D - Incorrect: Regular token structure does not describe Turing-machine simulation power.

Primary note proof

Study-note section AUT28

> An off-line TM has a read-only input tape and separate read/write work tape. Multitape, multihead, two-way-infinite-tape and nondeterministic TMs have the same language-recognition power as standard TMs, though simulation costs differ. Two unbounded stacks can simulate a tape. Turing completeness means ability to simulate general TM computation, not polynomial speed or literally using infinitely many cells in a finite run. The Church-Turing thesis connects effective procedures to this formal model; it is not a proved runtime theorem.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 20-21

> A context-free grammar
> can describe all regular languages and more, but they cannot describe all possible languages.

Automata and Complexity Theory module | PDF page 3 | lines 13-14

> A grammar defines a set of rules, and with the use of these rules, valid sentences in a language are
> generated.

Question sources

Compiler Design review questions · Question 45

Question 40 | Q01451

DFA is an abbreviation of.

A. Non Deterministic Finite set Automata

B. Deterministic Finite Automata

C. Non Deterministic Finite Automata

D. Deterministic Finite set Automata

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer B: Deterministic Finite Automata

DFA stands for deterministic finite automaton; deterministic specifies a unique next state for a given current state and input.

Why each choice is right or wrong

A - Incorrect: Adds nondeterministic and set, neither in the abbreviation.

B - Correct: This is the exact expansion of the abbreviation.

C - Incorrect: This is NFA, not DFA.

D - Incorrect: Inserts set, which is absent from the expansion.

Primary note proof

Study-note section AUT06

> A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0 \in Q$, accepting set $F \subseteq Q$ and total $\delta: Q \times \Sigma \to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition. F may be empty, recognizing $\varnothing$.

Study-note section AUT06

> Start at q_0 and consume the entire input. Accept iff the ending state is in F; visiting an accepting state earlier is insufficient. ε is accepted exactly when $q_0 \in F$. A nonaccepting sink loops on every symbol and cannot reach acceptance; it still finishes after finite input. Finite automata have finite memory encoded by states, not no state or no ability to remember any history.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 3-5

> DFA refers to deterministic finite automata. Deterministic refers to the uniqueness of the computation. In the DFA, the machine goes to one state only for a particular input character. DFA does not accept the null move.

Question sources

Compiler Design review questions · Question 59

Question 41 | Q01452

NFA is an abbreviation of.

A. Non Deterministic Finite set Automata

B. Deterministic Finite Automata

C. Non Deterministic Finite Automata

D. Deterministic Finite set Automata

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer C: Non Deterministic Finite Automata

NFA stands for nondeterministic finite automaton; it permits multiple possible next states and null moves.

Why each choice is right or wrong

A - Incorrect: Inserts set, which is absent from the abbreviation.

B - Incorrect: This expands DFA instead.

C - Correct: This is the exact expansion of the abbreviation.

D - Incorrect: This describes a deterministic model and inserts set.

Primary note proof

Study-note section AUT09

> A nondeterministic finite automaton (NFA) permits zero, one or many successors: $\delta:Q\times\Sigma\to P(Q)$. An ε-NFA additionally permits moves without consuming input, using domain $Q\times(\Sigma\cup\{\varepsilon\})$. Some books include ε moves in the term NFA; state the convention. An NFA accepts iff at least one finite path consumes the complete word and finishes in F, including permitted ε moves before/after symbols.

Study-note section AUT09

An NFA is not more expressive than a DFA: both recognize exactly REG. Nondeterminism can make a description smaller. A DFA is a special NFA after interpreting its unique successor as a singleton set.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 7-7

NFA stands for non-deterministic finite automata.

Question sources

Compiler Design review questions · Question 60

Question 42 | SM0018

Which branch of study deals with whether a problem can be solved at all or not, regardless of the resources required?

A. Computability theory

B. Automata theory

C. Set theory

D. Complexity theory

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer A: Computability theory

The resource-independent question is whether an algorithm can decide the problem at all. Computability concerns membership algorithms and the limits of what Turing machines compute; computational complexity concerns resource estimates.

Why each choice is right or wrong

A - Correct: Matches existence of an algorithm regardless of cost.

B - Incorrect: Automata concerns abstract machines; computability specifically asks the solvability question.

C - Incorrect: Computability, rather than set theory, studies algorithmic solvability.

D - Incorrect: Complexity concerns time and space resources once solving a problem is considered.

Primary note proof

Study-note section AUT-S01

Computability theory asks whether an effective algorithm can solve a problem at all. For a decision problem, a decision procedure must terminate with the correct yes-or-no result on every input.

Study-note section AUT-S01

Computational complexity theory studies the time and memory resources required to solve problems as functions of encoded input size. Automata theory studies abstract machine models and the languages they recognize; these subjects overlap but ask different questions.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 15 | lines 24-26

A set X is recursive if we have an algorithm to
determine whether a given element belongs to
X or not.

Automata and Complexity Theory module | PDF page 15 | lines 33-35

Algorithm analysis is an important part of a broader computational complexity theory, which provides theoretical estimates for the resources needed by any algorithm which solves a given computational problem.

Automata and Complexity Theory module | PDF page 11 | lines 7-7

A Turing machine can do everything that a real computer can do.

Question sources

MOE Computer Science Exit Exam · Question 5-2

National Computer Science Exit Exam · Question 38

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 246

Question 43 | SM0024

Let a Grammar G = ({S}, {a, b}, S, P), where {S} is set of variables, {a, b} is set of terminal symbols, S is start variable and P is set of productions. Then which one of the following is not an element of the language (L) (L) which is generated from G. Given:
P:S-- aSb,
S-A

A. aabb

B. ab

C. X

D. a

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer D: a is not generated.

The image restores S→aSb|λ and option C as λ, not the extracted X. AUT17 derives exactly $a^n b^n$ for n≥0. Two recursions generate aabb, one generates ab, and the base rule gives λ. A lone a has unequal counts and cannot arise. The readable source resolves the OCR damage rather than changing the grammar by assumption.

Why each choice is right or wrong

A - Incorrect: Two recursive pair additions followed by the empty rule derive aabb.

B - Incorrect: One application of S→aSb followed by S→λ derives ab.

C - Incorrect: The image reads lambda here, and the empty base production derives it directly.

D - Correct: A lone a has no matching b and is outside the generated equal-count language.

Primary note proof

Study-note section AUT17

> Use a recursive rule to enforce matched structure and a base rule to stop. S→aSb|ε generates exactly $a^n b^n$ for n≥0: n recursive steps add n a's on the left and n b's on the right. Derive aabb by S⇒aSb⇒aaSbb⇒aabb. For n≥1 replace ε by base ab.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string s∈(V∪Σ)*;

Automata and Complexity Theory module | PDF page 2 | lines 24-25

> A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Question sources

MOE Computer Science Exit Exam · Question 7-1

National Computer Science Exit Exam · Question 49

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Question 44 | SM0056

Which type of Turing machine has two tapes: one tape read-only and the other read-write?

A. Multi-head Turing machine

B. Off-line Turing machine

C. Non-deterministic Turing machine

D. Multi-dimensional Turing machine

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: An off-line Turing machine.

AUT28 directly defines an off-line TM as having a read-only input tape and a separate read/write work tape. The question gives that architecture. Multiple heads, nondeterministic transition choices and multidimensional tape geometry are independent variants; none defines this input/work separation. The number of tape roles alone does not imply those other variants.

Why each choice is right or wrong

A - Incorrect: Multi-head describes head multiplicity rather than this read-only input and writable work-tape separation.

B - Correct: The off-line model explicitly separates a read-only input tape from a read/write work tape.

C - Incorrect: Nondeterministic describes transition choices, not which tape permits writes.

D - Incorrect: Multidimensional describes tape geometry rather than the two tape roles specified here.

Primary note proof

Study-note section AUT28

> An off-line TM has a read-only input tape and separate read/write work tape. Multitape, multihead, two-way-infinite-tape and nondeterministic TMs have the same language-recognition power as standard TMs, though simulation costs differ. Two unbounded stacks can simulate a tape. Turing completeness means ability to simulate general TM computation, not polynomial speed or literally using infinitely many cells in a finite run. The Church-Turing thesis connects effective procedures to this formal model; it is not a proved runtime theorem.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 11 | lines 9-9

> It has a tape head that can read and write symbols and move around on the tape.

Automata and Complexity Theory module | PDF page 11 | lines 20-22

δ is the transition function, mapping (q, x) onto (q', y, D) where D denotes the direction of movement of R/W head: D = L or R according as the movement is to the left or right.

Question sources

MOE Computer Science Exit Exam · Question 16-3

National Computer Science Exit Exam · Question 36

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 248

Question 45 | SM0094

Given input alphabet Σ = {a, b, c, d} and empty string λ, which one of the following is true about Σ*?

A. $\Sigma^* = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4$

B. $\Sigma^* = \Sigma^+$

C. $\Sigma^* = \Sigma^+ \cup \{\lambda\}$

D. $\Sigma^* = \{\lambda\}$

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer C: $\Sigma^* = \Sigma^+ \cup \{\lambda\}$

Σ* consists of all finite symbol sequences including zero occurrences; Σ+ supplies the positive-length sequences. Since the alphabet has no empty symbol, exactly λ separates the two sets. Thus Σ*=Σ+∪{λ}.

Why each choice is right or wrong

A - Incorrect: Only lengths 1-4 are included; both λ and words of length 5 are missing.

B - Incorrect: Positive closure omits λ, unlike star.

C - Correct: Adds exactly the missing zero-length word.

D - Incorrect: Star also contains a and every other nonempty finite word over the alphabet.

Primary note proof

Study-note section AUT03

Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, $L^*=\cup n\geq 0 L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon \in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 5 | lines 25-25

> means one or more occurrence of x. It can generate {x, xx, xxx, xxxx,

Automata and Complexity Theory module | PDF page 2 | lines 24-25

> A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Question sources

MOE Computer Science Exit Exam · Question 27-1

National Computer Science Exit Exam · Question 55

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Question 46 | SM0095

Given input alphabet Σ = {a, b, c, d} and empty string λ, which one of the following is equivalent to Σ^0?

A. Σ^+

B. Σ^*

C. Σ

D. $\{\lambda\}$

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer D: The singleton set {λ}.

AUT03 defines Σ^n as all words of length n, including $\Sigma^0=\{\varepsilon\}$; AUT01 identifies λ with ε. There is exactly one length-zero word, so its set is {λ}. Σ, positive closure and Kleene closure contain nonempty words for the given nonempty alphabet and therefore are not equal to the zero power.

Why each choice is right or wrong

A - Incorrect: Σ^+ contains positive-length words over the alphabet and excludes the empty word.

B - Incorrect: Σ^* contains every finite-length word, not just the single empty word.

C - Incorrect: Σ contains four alphabet symbols, each forming a length-one word rather than a length-zero word.

D - Correct: λ denotes the empty word, and its singleton set is exactly the alphabet's zero power.

Primary note proof

Study-note section AUT03

> Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, $L^*=\bigcup n\geq 0L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Study-note section AUT01

> An alphabet Σ is a finite set of symbols. A word/string is a finite ordered sequence of symbols from Σ; a language is a set of words, $L\subseteq\Sigma^*$. An empty word ε, also written λ, has length zero. The empty language $\varnothing$ contains no words; $\{\varepsilon\}$ contains one.

Study-note section AUT01

> Length counts occurrences, not distinct symbols: $|caddba|=6$, so $|caddba|^2=36$.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 2 | lines 24-25

> A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Question sources

MOE Computer Science Exit Exam · Question 27-2

National Computer Science Exit Exam · Question 30

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Question 47 | SM0206

a - b, L state transition in a Turing Machine means,

A. read a, write b and move left

B. delete a, read b and move left

C. write a, read b and move left

D. replace 6 by a and move left

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: read a, write b and move left

In an edge label (a,b,L), the first symbol is the scanned symbol, the second is written in its cell, and L moves the head left. The finite control also moves to the destination state of the arrow.

Why each choice is right or wrong

A - Correct: Exactly follows read, write, move-left order.

B - Incorrect: The scanned symbol is a, and b is written, not read.

C - Incorrect: Reverses reading and writing.

D - Incorrect: Reverses the direction of replacement; the cell changes a to b.

Primary note proof

Study-note section AUT27

> δ(q,a)=(p,b,L) means read a in q, write b, move one cell left and enter p. Accept/reject halting states have no further moves under this convention.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 11 | lines 20-22

> δ is the transition function, mapping (q, x) onto (q', y, D) where D denotes the direction of movement of R/W head: D = L or R according as the movement is to the left or right.

Automata and Complexity Theory module | PDF page 11 | lines 9-9

> It has a tape head that can read and write symbols and move around on the tape.

Automata and Complexity Theory module | PDF page 14 | lines 8-8

> As a result the symbol is β written in the cell under the R/W head.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 2-2

Question 48 | SM0234

Which of the following IS NOT a component of a Turing machine?

A. Read-write head

B. Control Unit

C. Stack

D. Tape

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: Stack

The standard Turing machine uses an unbounded tape, a read/write head, and finite state control. Stack is the distinctive added memory of a PDA and is not a separate structural component required by this standard TM model.

Why each choice is right or wrong

A - Incorrect: Explicitly named as the reading/writing/moving device.

B - Incorrect: Finite states and transition rules provide control.

C - Correct: The standard TM model has tape storage, whereas the PDA description explicitly adds a stack.

D - Incorrect: The tape provides the TM's memory.

Primary note proof

Study-note section AUT27

> For a Turing machine (TM), use $M=(Q,\Sigma,\Gamma,\delta,q_0,q_{accept},q_{reject})$, with finite state/tape alphabets, $\Sigma\subseteq\Gamma$ and blank $\square\in\Gamma-\Sigma$. The blank symbol belongs to the tape alphabet Γ, not the input alphabet Σ. The standard machine has finite control, a read/write head and an unbounded tape. Input is initial tape content, not a separate hardware organ; no separate stack is required.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 11 | lines 9-9

> It has a tape head that can read and write symbols and move around on the tape.

Automata and Complexity Theory module | PDF page 11 | lines 20-22

> δ is the transition function, mapping (q, x) onto (q', y, D) where D denotes the direction of movement of R/W head: D = L or R according as the movement is to the left or right.

Automata and Complexity Theory module | PDF page 9 | lines 18-19

> Pushdown automata are nondeterministic finite state machines augmented with additional memory
> in the form of a stack,

Question sources

Computer Science Exit Exam · Sene 2016 · Question 11-2

Computer Science Exit Exam practice collection · Question 258

Question 49 | SM0246

In a grammar, if the production S -> lambda (where S is a non-terminal) exists, such a production is called:

A. Null production

B. Unit production

C. Secondary production

D. Sigma production

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer A: A null (epsilon) production.

AUT19 defines a null production as A→ε, while AUT01 identifies λ as another notation for the empty word. Substituting S for A gives the displayed S→λ rule. A unit production instead has one nonterminal on the right; secondary and sigma production are not the defined names for an empty right-hand side.

Why each choice is right or wrong

A - Correct: The right-hand side is the empty word, exactly the definition of a null production.

B - Incorrect: A unit production has one nonterminal on the right, not the empty word.

C - Incorrect: Secondary production is not the defined term for a rule deriving the empty word.

D - Incorrect: Sigma denotes an alphabet conventionally, not the null-production category described here.

Primary note proof

Study-note section AUT19

A null production is A→ε; a unit production is A→B with one variable.

Study-note section AUT01

An alphabet Σ is a finite set of symbols. A word/string is a finite ordered sequence of symbols from Σ; a language is a set of words, L⊆Σ*. An empty word ε, also written λ, has length zero. The empty language ∅ contains no words; {ε} contains one.

Study-note section AUT01

Length counts occurrences, not distinct symbols: |caddba|=6, so $|caddba|^2=36$.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 2 | lines 24-25

A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

R is a set of production rules where each production rule maps a variable to a string s∈(V∪Σ)∗;

Automata and Complexity Theory module | PDF page 9 | lines 15-16

Simplification essentially comprises the following steps; Reduction of CFG, Removal of Unit Productions, and Removal of Null Productions.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 15-2

Computer Science Exit Exam practice collection · Question 261

Question 50 | SM0256

Which string from the following is a member of the language represented by the regular expression a*(b+c)?

A. aab

B. bc

C. aaa

D. aaabc

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer A: aab

a* permits any number of initial a symbols; (b+c) then contributes exactly one b or c. aab factors as aa followed by b. No further suffix is allowed after that final symbol.

Why each choice is right or wrong

A - Correct: aa is in a*, followed by the permitted b.

B - Incorrect: Contains two symbols from the final alternative; the a* prefix cannot consume b.

C - Incorrect: Lacks the compulsory final b or c.

D - Incorrect: After aa a and b there is an extra c; the expression allows only one final alternative.

Primary note proof

Study-note section AUT12

> Classical expressions use ∅, ε, symbols, union, concatenation and star. A union B is written A|B or A+B; postfix r^+ means one-or-more r repetitions and is a different use of +. Parenthesize when conventions might clash. Star binds tightest, concatenation next, union last. r? means r|ε; r^+=rr*.

Study-note section AUT12

> (a|b)*a denotes words ending a; 0*10* denotes exactly one 1; a*(b|c) gives zero-or-more a's then exactly one b or c, so aab belongs but aaa and aaabc do not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

> A language is said to be a Regular language if and only if some finite state machine recognizes it

Question sources

Computer Science Exit Exam · Sene 2016 · Question 18-2

Computer Science Exit Exam practice collection · Question 259

Question 51 | SM0303

Which class of languages is most suited to be represented by pushdown automata?

A. Context-free language

B. Regular language

C. Recursively enumerable language

D. Unrestricted language

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Context-free language

PDAs and CFGs characterize the same family of context-free languages. Regular languages are included, but the question most suited asks for the full family rather than a subclass.

Why each choice is right or wrong

A - Correct: Exactly the class characterized by PDAs.

B - Incorrect: A smaller included family, already recognizable by finite automata.

C - Incorrect: The hierarchy places RE at the larger Turing machine level.

D - Incorrect: Unrestricted languages are not the PDA/CFG class.

Primary note proof

Study-note section AUT21

> A nondeterministic pushdown automaton (PDA) adds a stack to finite control: $(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$. Its transition relation maps (state,input symbol or ε,stack top) to possible (new state,replacement stack string) pairs. Fix which end denotes the top before tracing. Replacing X by X leaves the stack unchanged; ε input means consume no symbol, while replacing X by ε pops it. These are different uses of ε.

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use $REG \subsetneq CFL \subsetneq CSL \subsetneq DECIDABLE \subsetneq RE$, under the usual convention allowing $S \to \varepsilon$ with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 9 | lines 23-23

> Pushdown automata accept context-free languages, which include the set of regular languages.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 33-3

Computer Science Exit Exam practice collection · Question 257

Question 52 | AM0744

Which statement correctly describes the generation and recognition roles of formal grammars and automata?

A. Formal grammars generate formal languages, and automata recognize those languages.

B. Formal languages recognize automata, which are generated by formal grammars.

C. Formal grammars generate automata, which recognize formal languages.

D. Formal grammars recognize formal languages, which generate automata.

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer A: Formal grammars generate formal languages, and automata recognize those languages.

A formal grammar generates terminal words, collectively its language. An automaton processes words and accepts exactly those in the language it recognizes. Thus languages are generated by grammars and recognized by automata. The relationship concerns the same set of strings, not a claim that automata recognize grammar descriptions or that languages themselves perform machine operations. Option A states those roles explicitly.

Why each choice is right or wrong

A - Correct: It correctly assigns language generation to grammars and recognition to automata.

B - Incorrect: Languages do not perform recognition or generate machines.

C - Incorrect: Grammars generate word languages rather than automata in this relationship.

D - Incorrect: Grammars generate languages; languages do not generate automata.

Primary note proof

Study-note section AUT04

> Write G=(V,Σ,P,S): finite variables V, terminals Σ disjoint from V, productions P and start variable S∈V. A production replaces its left side by its right side. The language is L(G)={w∈Σ*:S⇒*w}. A recognizer processes a word and accepts exactly the language members. Grammars generate languages; automata recognize languages, not grammars.

Study-note section AUT04

> Two grammars are equivalent when their generated languages are equal, regardless of rule counts or variable names. One common word, equal alphabets or one-way inclusion is insufficient.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 3 | lines 13-14

A grammar defines a set of rules, and with the use of these rules, valid sentences in a language are
generated.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

A language is said to be a Regular language if and only if some finite state machine recognizes it

Automata and Complexity Theory module | PDF page 10 | lines 6-6

Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 744

Question 53 | AM0745

Which offered item is a precisely specified formal language, rather than a natural-language name?

A. English

B. Spanish

C. L={w∈{a,b}* | w ends in a}, denoted by (a|b)*a

D. French

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer C: L={w∈{a,b}* | w ends in a}, denoted by (a|b)*a

A formal language is a set of strings with precisely defined membership. Option C gives the language of all words over {a,b} that end in a, denoted by (a|b)*a. English, Spanish and French are natural-language names in this comparison. A regular expression alone names a specification notation; explicitly naming its denoted word set provides the actual language example.

Why each choice is right or wrong

A - Incorrect: English is a natural-language name.

B - Incorrect: Spanish is a natural-language name.

C - Correct: This specifies a set of words over a fixed alphabet with an exact ending condition.

D - Incorrect: French is a natural-language name.

Primary note proof

Study-note section AUT04

Write G=(V,Σ,P,S): finite variables V, terminals Σ disjoint from V, productions P and start variable S∈V. A production replaces its left side by its right side. The language is L(G)={w∈Σ*:S⇒*w}. A

recognizer processes a word and accepts exactly the language members. Grammars generate languages; automata recognize languages, not grammars.

Study-note section AUT04

Two grammars are equivalent when their generated languages are equal, regardless of rule counts or variable names. One common word, equal alphabets or one-way inclusion is insufficient.

Study-note section AUT12

Classical expressions use ∅, ε, symbols, union, concatenation and star. A union B is written A|B or A+B; postfix r^+ means one-or-more r repetitions and is a different use of +. Parenthesize when conventions might clash. Star binds tightest, concatenation next, union last. r? means r|ε; r^+=rr*.

Study-note section AUT12

(a|b)*a denotes words ending a; 0*10* denotes exactly one 1; a*(b|c) gives zero-or-more a's then exactly one b or c, so aab belongs but aaa and aaabc do not.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 3 | lines 13-14

A grammar defines a set of rules, and with the use of these rules, valid sentences in a language are
generated.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

A language is said to be a Regular language if and only if some finite state machine recognizes it

Question sources

Computer Science MCQ practice collection · Question 745

Question 54 | AM0746

Which is the least expressive listed machine family that recognizes every regular language?

A. Turing machine

B. Pushdown automaton

C. Finite automaton

D. Nondeterministic Turing machine

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: Finite automaton

Finite automata recognize exactly the regular languages. Pushdown automata and Turing machines can also recognize regular languages, but their families recognize strictly more languages and are not the least sufficient choice. The nondeterministic Turing-machine alternative is likewise stronger than finite automata. This compares expressive power across machine families, rather than implementation size or execution speed.

Why each choice is right or wrong

A - Incorrect: Turing machines recognize a strictly larger family than REG.

B - Incorrect: Pushdown automata recognize all CFLs, exceeding REG.

C - Correct: Finite automata characterize REG exactly and are least sufficient here.

D - Incorrect: Nondeterministic Turing machines exceed finite-automaton recognition power.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

> A language is said to be a Regular language if and only if some finite state machine recognizes it

Automata and Complexity Theory module | PDF page 9 | lines 23-23

> Pushdown automata accept context-free languages, which include the set of regular languages.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 746

Question 55 | AM0747

Which statements follow from the Chomsky hierarchy and decidability? Select all that apply.

A. Context-free languages are a subset of regular languages.

B. Context-sensitive languages are a subset of recursive languages.

C. Regular languages are a subset of recursive languages.

D. Context-free languages are a subset of recursive languages.

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B, C, D: Context-sensitive languages are a subset of recursive languages.; Regular languages are a subset of recursive languages.; Context-free languages are a subset of recursive languages.

The hierarchy is REG⊊CFL⊊CSL⊊DECIDABLE⊊RE under the stated ε convention. Recursive means decidable, so context-sensitive languages are recursive, and transitivity gives the same conclusion for regular and context-free languages. A reverses the first strict inclusion: every regular language is context-free, but not every context-free language is regular. Therefore the three recursive-containment statements are all true.

Why each choice is right or wrong

A - Incorrect: The REG/CFL inclusion is reversed.

B - Correct: CSL is contained in the decidable or recursive family.

C - Correct: REG is contained in CSL and therefore in the recursive family.

D - Correct: CFL is contained in CSL and therefore in the recursive family.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 8 | lines 20-21

> A context-free grammar
> can describe all regular languages and more, but they cannot describe all possible languages.

Automata and Complexity Theory module | PDF page 15 | lines 24-26

> A set X is recursive if we have an algorithm to
> determine whether a given element belongs to
> X or not.

Automata and Complexity Theory module | PDF page 15 | lines 30-30

> All recursive set is recursively enumerable.

Question sources

Computer Science MCQ practice collection · Question 747

Question 56 | AM0748

Which statement is true of ordinary DFAs? In D, the claim concerns every context-free language.

A. DFAs can recognize all regular languages.

B. DFAs have multiple possible transitions for each input symbol.

C. DFAs can have empty transitions.

D. DFAs can recognize every context-free language.

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer A: DFAs can recognize all regular languages.

Ordinary DFAs recognize exactly the regular languages. Each state/input pair has one next state, and transitions consume an input symbol rather than ε. Some context-free languages are regular, but others, such as $a^n b^n$, are not and cannot be recognized by a DFA. Making D explicitly universal removes its weaker existential reading and leaves A as the unique true statement.

Why each choice is right or wrong

A - Correct: Every regular language has a DFA.

B - Incorrect: A DFA has a unique successor for each state/symbol pair.

C - Incorrect: Ordinary DFAs do not have ε moves.

D - Incorrect: Nonregular CFLs such as $a^n b^n$ cannot be recognized by DFAs.

Primary note proof

Study-note section AUT06

A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0 \in Q$, accepting set $F \subseteq Q$ and total $\delta: Q \times \Sigma \to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition.

Study-note section AUT09

An NFA is not more expressive than a DFA: both recognize exactly REG.

Study-note section AUT15

For $L=\{a^n b^n : n \geq 0\}$, choose $a^p b^p$. Every allowed $y=a^k$ with $k \geq 1$; pumping $i=0$ leaves fewer a's than b's. The contradiction excludes every DFA.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 3-5

DFA refers to deterministic finite automata. Deterministic refers to the uniqueness of the computation. In the DFA, the machine goes to one state only for a particular input character. DFA does not accept the null move.

Automata and Complexity Theory module | PDF page 7 | lines 16-16

A language is said to be a Regular language if and only if some finite state machine recognizes it

Automata and Complexity Theory module | PDF page 8 | lines 20-21

A context-free grammar
can describe all regular languages and more, but they cannot describe all possible languages.

Question sources

Computer Science MCQ practice collection · Question 748

Question 57 | AM0749

Which listed automaton family has recognition power strictly between finite automata and Turing machines and uses one unbounded stack as its auxiliary memory?

A. Pushdown automaton

B. Non-deterministic finite automaton

C. Turing machine

D. Nondeterministic linear bounded automaton

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Pushdown automaton

A nondeterministic PDA adds one stack to finite control and recognizes the context-free languages. This family strictly contains REG and lies strictly below Turing-recognizable languages. NFAs have the same power as DFAs, while Turing machines are the upper endpoint. A nondeterministic LBA also lies between those endpoints, but uses bounded tape rather than the specified single-stack model.

Why each choice is right or wrong

A - Correct: A PDA has one unbounded stack and the required intermediate expressive power.

B - Incorrect: An NFA has no stack and recognizes only REG.

C - Incorrect: A TM is the upper endpoint, not strictly below it.

D - Incorrect: An LBA uses linearly bounded tape rather than the specified single stack.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Study-note section AUT21

> A nondeterministic pushdown automaton (PDA) adds a stack to finite control: $(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$.

Study-note section AUT25

> A linear bounded automaton (LBA) is a TM with O(n) tape cells on length-n input; the standard CSL characterization uses nondeterministic LBAs.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 9 | lines 21-21

> machines-that can do more than a finite state machine, but less than a Turing machine.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 749

Question 58 | AM0751

Which statements about NP-complete decision problems are true? Select all that apply. In C, exponential time is an upper bound.

A. NP-complete problems are only solvable by nondeterministic algorithms.

B. NP-complete problems are a subset of NP problems.

C. NP-complete problems can be solved in exponential time.

D. NP-complete problems have no practical applications.

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer B, C: NP-complete problems are a subset of NP problems.; NP-complete problems can be solved in exponential time.

NP-completeness includes membership in NP and hardness for every NP language under polynomial reductions. Polynomial-size certificates permit finite deterministic exhaustive search, giving a 2^poly(n) upper bound. Thus B and C are true together. This does not show that exponential time is necessary. SAT, coloring and tour-decision formulations illustrate meaningful problems, so the blanket denial of applications is also false.

Why each choice is right or wrong

A - Incorrect: Deterministic exhaustive certificate search also decides these problems.

B - Correct: Membership in NP is part of the definition.

C - Correct: Enumerating polynomial-size certificates gives a deterministic exponential-time upper bound.

D - Incorrect: The note gives substantive SAT, coloring and tour-decision examples.

Primary note proof

Study-note section AUT36

> X is NP-hard if every A∈NP satisfies A≤pX. It need not belong to NP and can even be undecidable. X is NP-complete if X∈NP and X is NP-hard. NP-complete languages lie inside NP and can be decided by finite exhaustive certificate search in exponential time. They are not known to require exponential time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 17 | lines 27-28

> P problems are set of problems which can be solved in
> polynomial time by deterministic algorithms.

Automata and Complexity Theory module | PDF page 17 | lines 29-30

> NP problems are the problems which can be solved in non-deterministic polynomial time.

Automata and Complexity Theory module | PDF page 17 | lines 33-35

> The problem belongs to NP, if it‘s easy to check a solution that may have been very tedious to find.

Question sources

Computer Science MCQ practice collection · Question 751

Question 59 | AM0752

Which of the following complexity classes represents the hardest problems in NP?

A. P

B. NP-complete

C. NP-hard

D. PSPACE

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer B: NP-complete: the problems both in NP and NP-hard.

AUT36 defines NP-complete as NP membership together with hardness for every NP problem under polynomial reductions. That is the precise class described by hardest problems in NP. NP-hard alone can extend beyond NP, even to undecidable problems; PSPACE is a larger resource class, and P does not impose the required hardness.

Why each choice is right or wrong

A - Incorrect: P means deterministic polynomial time and does not define hardness for every NP language.

B - Correct: NP-completeness combines being in NP with being at least as hard as every NP problem.

C - Incorrect: NP-hardness alone does not ensure membership in NP, so it is broader than the class requested.

D - Incorrect: PSPACE is defined by polynomial space and can contain languages outside the specified NP-complete class.

Primary note proof

Study-note section AUT36

X is NP-hard if every A∈NP satisfies A≤pX. It need not belong to NP and can even be undecidable. X is NP-complete if X∈NP and X is NP-hard. NP-complete languages lie inside NP and can be decided by finite exhaustive certificate search in exponential time. They are not known to require exponential time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 17 | lines 29-30

NP problems are the problems which can be solved in non-deterministic polynomial time.

Automata and Complexity Theory module | PDF page 17 | lines 33-35

The problem belongs to NP, if it‘s easy to check a solution that may have been very tedious to find.

Question sources

Computer Science MCQ practice collection · Question 752

Question 60 | AM0753

Which of the following is true regarding the halting problem?

A. The halting problem can be solved by a Turing machine.

B. The halting problem is decidable for all programs.

C. The halting problem is an undecidable problem.

D. The halting problem is only relevant to regular languages.

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer C: The general halting problem is undecidable.

AUT30 constructs a machine that contradicts a hypothetical universal halting decider when run on its own description. Hence no decider answers halting correctly for all machine/input pairs. HALT is recognizable by simulation, but that process need not terminate on nonhalting instances; recognizing some yes-instances is not solving the general decision problem.

Why each choice is right or wrong

A - Incorrect: Simulation recognizes halting yes-instances but no TM decides the general halting problem on every input.

B - Incorrect: The diagonal argument rules out a decider correct for all programs and inputs.

C - Correct: Undecidability is the exact result proved by the halting diagonalization argument in AUT30.

D - Incorrect: HALT concerns arbitrary computations and is not confined to regular languages or finite automata.

Primary note proof

Study-note section AUT30

> HALT={⟨M,w⟩:M halts on w} is recognizable: simulate and accept upon any halt. Suppose a decider H determines halting. Construct D(x) that loops if H(x,x) says halt, otherwise halts. Running D on its own description contradicts H in both cases. Thus HALT is undecidable and witnesses RE beyond DECIDABLE. Its complement is not RE, or both recognizers would yield a decider.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 14 | lines 19-20

> -It Is impossible to construct a single algorithm
> that always leads to a correct yes-or-no answer.

Automata and Complexity Theory module | PDF page 15 | lines 24-26

> A set X is recursive if we have an algorithm to
> determine whether a given element belongs to
> X or not.

Question sources

Computer Science MCQ practice collection · Question 753

Question 61 | AM0754

Which of the following is a regular language?

A. {anbn | n ≥ 0}

B. {anbncn | n ≥ 0}

C. {an | n is a prime number}

D. {anb | n ≥ 0}

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer D: {anb | n ≥ 0}

Reading the powers as a^n b^n, a^n b^n c^n, a^n for prime n, and a^n b. D is a*b, a regular expression. A and B fail the pumping lemma's condition: choose a^p b^p (or a^p b^p c^p); any nonempty pumped block in the first p positions consists only of a, so pumping changes only its count. For C choose a prime q at least p. If the pumped block has k≥1 a symbols, pumping with i=q+1 gives length q+qk=q(k+1), composite. Thus no pumping decomposition works. The argument uses the arithmetic fact that arbitrarily large primes exist.

Why each choice is right or wrong

A - Incorrect: Pumping only a symbols destroys equality with the b count.

B - Incorrect: Pumping only a symbols destroys equality with both later blocks.

C - Incorrect: Pumping the unary block q+1 times makes a composite length q(k+1).

D - Correct: a*b produces any number of a symbols followed by exactly one b.

Primary note proof

Study-note section AUT26

> a*b* and {$a^n b$:n≥0} are regular. $a^n b^n$ is CFL but not regular. Over a two-symbol alphabet, ww^R is CFL, ww is not CFL; both are decidable. $a^n b^n c^n$ is CSL but not CFL, and therefore decidable. Every finite language is regular. A stronger grammar or TM can still recognize any of these lower-family languages.

Study-note section AUT26

> Unary prime length {a^p:p prime} is nonregular: choose prime q above pumping length p; if pump piece has length k≥1, using exponent q+1 gives length q(k+1), composite. Every unary CFL is regular, so this language is not CFL either. It is nevertheless context-sensitive: count the input in bounded workspace and test divisors. The balanced-three-count, prime-length, palindrome and $a^n b$ examples above are all recursive languages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 12-12

> (1) |uv| ≤ n (2) |v| ≥ 1 (3) for all i ≥ 0: uviw ∈ L

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string s∈(V∪Σ)*;

Question sources

Computer Science MCQ practice collection · Question 754

Question 62 | AM0755

Which is the least expressive listed grammar family capable of generating every context-free language?

A. Regular grammar

B. Context-free grammar

C. Context-sensitive grammar

D. Unrestricted grammar

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: Context-free grammar

Context-free grammars generate exactly the context-free languages. Regular grammars are insufficient because CFL contains nonregular languages such as a^nb^n. Context-sensitive and unrestricted grammars can also generate CFLs under the stated ε convention, but their expressive power is greater than required. The least-family qualifier distinguishes the exact defining family from stronger families that merely have the same capability as a subset.

Why each choice is right or wrong

A - Incorrect: Regular grammars fail for nonregular CFLs such as a^nb^n.

B - Correct: Context-free grammars cover exactly the required family.

C - Incorrect: Context-sensitive grammars suffice but are stronger than required.

D - Incorrect: Unrestricted grammars suffice but are stronger than required.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 20-21

A context-free grammar
can describe all regular languages and more, but they cannot describe all possible languages.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 755

Question 63 | AM0756

Under the usual isolated-start ε convention, which statements about CFLs and CSLs are true? Select all that apply.

A. Every context-sensitive language is also a context-free language.

B. Every context-free language is also a context-sensitive language.

C. Context-free languages are a strict subset of context-sensitive languages.

D. Context-free languages are unrelated to context-sensitive languages.

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B, C: Every context-free language is also a context-sensitive language.; Context-free languages are a strict subset of context-sensitive languages.

Every context-free language is context-sensitive under the stated ε convention. The containment is strict: $a^n b^n c^n$ is context-sensitive but not context-free. Hence both the inclusive statement B and the stronger proper-subset statement C are true. A reverses the containment, and D denies an established relationship. Selecting the stronger statement alone would incorrectly discard a weaker statement that it logically entails.

Why each choice is right or wrong

A - Incorrect: The three-equal-count language is a CSL outside CFL.

B - Correct: Every CFL belongs to CSL.

C - Correct: CSL contains additional languages, making the inclusion strict.

D - Incorrect: The two families are related by strict containment.

Primary note proof

Study-note section AUT05

Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 8 | lines 20-21

> A context-free grammar
> can describe all regular languages and more, but they cannot describe all possible languages.

Question sources

Computer Science MCQ practice collection · Question 756

Question 64 | AM0757

Which listed machine family can recognize every context-sensitive language? Interpret D as a nondeterministic finite automaton.

A. Finite automaton

B. Pushdown automaton

C. Turing machine

D. Nondeterministic finite automaton

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: Turing machine

Context-sensitive languages are recognized by nondeterministic linear bounded automata. An unrestricted Turing machine can simulate their bounded-tape computations, so it can recognize every CSL. Finite automata and NFAs recognize only regular languages, and ordinary PDAs recognize only context-free languages. Those weaker families cannot cover the three-equal-count CSL, so they fail the universal capability requirement.

Why each choice is right or wrong

A - Incorrect: Finite automata cannot recognize nonregular CSLs.

B - Incorrect: PDAs cannot recognize CSLs outside CFL.

C - Correct: A TM can simulate the LBA model for every CSL.

D - Incorrect: NFAs recognize only REG, not every CSL.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Study-note section AUT25

> A context-sensitive rule αAβ→αγβ replaces A in context, with γ nonempty. Equivalently, noncontracting grammar presentations preserve/increase total length, with the isolated ε exception. The condition is on the complete sides, not an unrelated comparison |β|≥|α|.

Study-note section AUT25

> A linear bounded automaton (LBA) is a TM with O(n) tape cells on length-n input; the standard CSL characterization uses nondeterministic LBAs. For $a^nb^nc^n$ first verify a*b*c* order, repeatedly mark one unmarked a, b and c, and reject missing/excess symbols.

Study-note section AUT25

> Thus every CSL is decidable.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 11 | lines 7-7

> A Turing machine can do everything that a real computer can do.

Question sources

Computer Science MCQ practice collection · Question 757

Question 65 | AM0758

Which listed term names exactly the recursively enumerable language family, rather than a proper subfamily?

A. Regular language

B. Context-free language

C. Context-sensitive language

D. Turing-recognizable language

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer D: Turing-recognizable language

Recursively enumerable, Turing-recognizable and semidecidable are names for the same language family. Regular, context-free and context-sensitive languages are all contained in RE, but each is a proper subfamily. The exact-family wording therefore singles out Turing-recognizable. Recognizers must halt and accept members; on a nonmember they may either reject or loop, unlike deciders which halt on every input.

Why each choice is right or wrong

A - Incorrect: REG is a proper subfamily of RE.

B - Incorrect: CFL is a proper subfamily of RE.

C - Incorrect: CSL is a proper subfamily of RE.

D - Correct: Turing-recognizable is an exact synonym for RE.

Primary note proof

Study-note section AUT29

> A decider halts on every input and accepts exactly its language; these languages are recursive/decidable. A recognizer must halt and accept every member, but on a nonmember may halt-reject or loop. Its languages are recursively enumerable (RE), Turing-recognizable or semidecidable. Looping is nonacceptance, not a rejecting halt. An undecidable language may still have a recognizer.

Study-note section AUT29

> Every decidable language is RE, and the containment is strict.

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 15 | lines 30-30

All recursive set is recursively enumerable.

Question sources

Computer Science MCQ practice collection · Question 758

Question 66 | AM0759

Which class of languages is the most powerful in terms of generative power?

A. Regular languages

B. Context-free languages

C. Context-sensitive languages

D. Recursively enumerable languages

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer D: Recursively enumerable languages

Reading most powerful as largest generative family in the traditional hierarchy, the outermost region is recursively enumerable. It properly surrounds context-sensitive, context-free and regular classes.

Why each choice is right or wrong

A - Incorrect: Innermost family, hence smallest listed.

B - Incorrect: Contains regular but is inside context-sensitive.

C - Incorrect: Larger than CFL but still inside RE.

D - Correct: Outermost family in the diagram.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 759

Question 67 | AM0760

Which statements about recursive and recursively enumerable languages are true? Select all that apply.

A. Every recursive language is also a recursively enumerable language.

B. Every recursively enumerable language is also a recursive language.

C. Recursive languages are a strict subset of recursively enumerable languages.

D. Recursive languages are unrelated to recursively enumerable languages.

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer A, C: Every recursive language is also a recursively enumerable language.; Recursive languages are a strict subset of recursively enumerable languages.

Every decider is also a recognizer, so each recursive language is recursively enumerable. The containment is proper: the halting language can be recognized by simulation but cannot be decided on every input. Thus both the inclusion statement and the strict-subset statement are true. The converse would turn every recognizer into a total decider, contradicting the halting example; describing the two families as unrelated also contradicts their inclusion.

Why each choice is right or wrong

A - Correct: A decider recognizes its language and halts on every input, establishing inclusion.

B - Incorrect: The recognizable but undecidable halting language is a counterexample.

C - Correct: The halting language belongs to RE but not to the recursive family, making inclusion strict.

D - Incorrect: The families are related by proper containment.

Primary note proof

Study-note section AUT29

> A decider halts on every input and accepts exactly its language; these languages are recursive/decidable. A recognizer must halt and accept every member, but on a nonmember may halt-reject or loop. Its languages are recursively enumerable (RE), Turing-recognizable or semidecidable. Looping is nonacceptance, not a rejecting halt. An undecidable language may still have a recognizer.

Study-note section AUT29

> Every decidable language is RE, and the containment is strict.

Study-note section AUT30

> HALT={⟨M,w⟩:M halts on w} is recognizable: simulate and accept upon any halt. Suppose a decider H determines halting. Construct D(x) that loops if H(x,x) says halt, otherwise halts. Running D on its own description contradicts H in both cases. Thus HALT is undecidable and

> witnesses RE beyond DECIDABLE. Its complement is not RE, or both recognizers would yield a decider.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 15 | lines 30-30

> All recursive set is recursively enumerable.

Automata and Complexity Theory module | PDF page 15 | lines 24-26

> A set X is recursive if we have an algorithm to
> determine whether a given element belongs to
> X or not.

Question sources

Computer Science MCQ practice collection · Question 760

Question 68 | AM0761

Which of the following is an example of an undecidable problem?

A. Checking if a given number is prime

B. Sorting a list of numbers in ascending order

C. Solving a system of linear equations

D. Determining if a Turing machine halts on a given input

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer D: Determining whether an arbitrary TM halts on its given input.

AUT30 proves HALT undecidable, directly matching D. The other choices describe ordinary finite mathematical computation: primality can test finitely many candidate divisors, a finite list can be sorted, and an exact finite linear system can be processed by elimination. Their cost can vary, but no general undecidability follows from those ordinary formulations.

Why each choice is right or wrong

A - Incorrect: For an integer, testing finitely many candidate divisors decides primality; AUT26 uses bounded divisor testing.

B - Incorrect: A finite numeric list can be sorted by a terminating comparison procedure.

C - Incorrect: An ordinarily encoded finite linear system can be decided and solved by finite elimination steps.

D - Correct: This is the general halting decision problem proved undecidable in AUT30.

Primary note proof

Study-note section AUT30

Thus HALT is undecidable and witnesses RE beyond DECIDABLE.

Study-note section AUT26

It is nevertheless context-sensitive: count the input in bounded workspace and test divisors.

Study-note section AUT-S04

Sorting a finite numeric list is a terminating computation. Quicksort, for example, chooses a pivot, partitions the other entries and recursively sorts the smaller sublists. Each recursive subproblem has fewer entries. A large running time does not make the task undecidable.

Study-note section AUT-S04

A finite linear system with exactly represented rational coefficients can be processed by Gaussian elimination and back substitution. Row exchanges handle unsuitable pivots; row reduction detects inconsistency or leaves free variables describing multiple solutions. These finite algebraic procedures differ from a general halting decider, which does not exist.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 14 | lines 19-20

-It Is impossible to construct a single algorithm
that always leads to a correct yes-or-no answer.

Automata and Complexity Theory module | PDF page 15 | lines 24-26

A set X is recursive if we have an algorithm to
determine whether a given element belongs to
X or not.

Question sources

Computer Science MCQ practice collection · Question 761

Question 69 | AM0762

Which offered language is nonrecursive (undecidable)? Use finite standard encodings and a fixed finite alphabet for mirror words.

A. HALT={⟨M,w⟩ | M halts on w}

B. {an | n is a prime number}

C. {wwR | w is any string}

D. {anb | n ≥ 0}

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer A: HALT={⟨M,w⟩ | M halts on w}

The halting language HALT contains encoded machine/input pairs for which the machine eventually halts. The diagonal argument in the note proves that no algorithm decides HALT on every input. The remaining examples are decidable: unary primality uses finite divisor testing, mirror words can be compared finitely, and a^nb is regular. Thus nonregularity or an expensive test alone does not imply nonrecursiveness.

Why each choice is right or wrong

A - Correct: HALT is recognizable but undecidable by the diagonal argument.

B - Incorrect: Finite counting and divisor testing decide unary prime length.

C - Incorrect: A finite mirror comparison decides palindrome membership.

D - Incorrect: The language a^nb is regular and hence decidable.

Primary note proof

Study-note section AUT30

> HALT={⟨M,w⟩:M halts on w} is recognizable: simulate and accept upon any halt. Suppose a decider H determines halting. Construct D(x) that loops if H(x,x) says halt, otherwise halts. Running D on its own description contradicts H in both cases. Thus HALT is undecidable and witnesses RE beyond DECIDABLE. Its complement is not RE, or both recognizers would yield a decider.

Study-note section AUT26

> a*b* and $\{a^nb:n\geq 0\}$ are regular. a^nb^n is CFL but not regular. Over a two-symbol alphabet, ww^R is CFL, ww is not CFL; both are decidable. $a^nb^nc^n$ is CSL but not CFL, and therefore decidable. Every finite language is regular. A stronger grammar or TM can still recognize any of these lower-family languages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 15 | lines 24-26

> A set X is recursive if we have an algorithm to
> determine whether a given element belongs to
> X or not.

Automata and Complexity Theory module | PDF page 11 | lines 7-7

> A Turing machine can do everything that a real computer can do.

Question sources

Computer Science MCQ practice collection · Question 762

Question 70 | AM0763

At the level of generated language families, what criterion orders the Chomsky hierarchy?

A. It categorizes formal languages based on their computational complexity.

B. It categorizes formal languages based on their expressive power.

C. It categorizes formal languages based on their alphabet size.

D. It categorizes languages by the surface structure of individual input strings.

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: It categorizes formal languages based on their expressive power.

The hierarchy orders families by expressive power: regular languages form a proper subfamily of CFLs, then CSLs, then the stronger computable families. Grammar production restrictions define the corresponding types, but the resulting language-family comparison concerns generative capability. It is not a ranking of algorithm runtimes, alphabet sizes, or surface appearance of particular strings. That scope makes B the unique criterion.

Why each choice is right or wrong

A - Incorrect: The hierarchy is not a runtime or workspace ranking.

B - Correct: It compares the language families that the models can describe.

C - Incorrect: Alphabet size does not determine the family.

D - Incorrect: Surface structure of individual strings is not the language-family ordering criterion.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 3 | lines 13-14

A grammar defines a set of rules, and with the use of these rules, valid sentences in a language are
generated.

Question sources

Computer Science MCQ practice collection · Question 763

Question 71 | AM0764

What is the least expressive listed grammar family that generates $L=\{a^nb^n \mid n\geq 0\}$? Use the usual isolated-start ε convention.

A. Regular grammar

B. Context-free grammar

C. Context-sensitive grammar

D. Unrestricted grammar

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer B: Context-free grammar

The CFG $S\to aSb|\varepsilon$ generates exactly equal blocks of a and b: every recursive step adds one of each and the base supplies n=0. The pumping argument rules out regularity. Context-sensitive and unrestricted grammars can also generate this language, but are stronger than necessary under the ε convention. The least expressive sufficient family is therefore context-free.

Why each choice is right or wrong

A - Incorrect: Regular grammars cannot enforce these unbounded equal counts.

B - Correct: The displayed matched recursive CFG generates the language.

C - Incorrect: Context-sensitive grammars are stronger than necessary.

D - Incorrect: Unrestricted grammars are stronger than necessary.

Primary note proof

Study-note section AUT17

Use a recursive rule to enforce matched structure and a base rule to stop. $S\to aSb|\varepsilon$ generates exactly a^nb^n for $n\geq 0$: n recursive steps add n a's on the left and n b's on the right. Derive aabb by $S\Rightarrow aSb\Rightarrow aaSbb\Rightarrow aabb$. For $n\geq 1$ replace ε by base ab.

Study-note section AUT15

If L is regular, some $p\geq 1$ satisfies: every $s\in L$ with $|s|\geq p$ has a split $s=xyz$, $|xy|\leq p$, $|y|\geq 1$, and $xy^iz\in L$ for every $i\geq 0$.

Study-note section AUT15

For L={$a^n b^n$:n≥0}, choose $a^p b^p$. Every allowed y=a^k with k≥1; pumping i=0 leaves fewer a's than b's. The contradiction excludes every DFA.

Study-note section AUT05

Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

R is a set of production rules where each production rule maps a variable to a string s∈(V∪Σ)∗;

Automata and Complexity Theory module | PDF page 8 | lines 12-12

(1) |uv| ≤ n (2) |v| ≥ 1 (3) for all i ≥ 0: uviw ∈ L

Automata and Complexity Theory module | PDF page 10 | lines 6-6

Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 764

Question 72 | AM0765

Which listed language is context-sensitive but NOT context-free? Interpret the notation as powers and use the usual ε convention.

A. {$a^n b^n$ | n≥0}

B. {$a^n b^m$ | n,m≥0}

C. {a^n | n is prime}

D. {$a^n b$ | n≥0}

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: {a^n | n is prime}

Unary prime length is context-sensitive because bounded-workspace counting and divisor testing decide it. It is not regular by the prime-length pumping argument, and every unary CFL is regular; consequently it is not context-free. In contrast, a^nb^n is context-free, while independently chosen a/b counts and any number of a followed by one b are regular. These weaker examples are excluded by the explicit non-CFL condition.

Why each choice is right or wrong

A - Incorrect: Equal two-block counts form a CFL.

B - Incorrect: Independent counts give the regular language a*b*.

C - Correct: Unary prime length is CSL but not CFL.

D - Incorrect: Any number of a followed by one b is regular.

Primary note proof

Study-note section AUT26

> a*b* and {a^nb:n≥0} are regular. a^nb^n is CFL but not regular. Over a two-symbol alphabet, ww^R is CFL, ww is not CFL; both are decidable. $a^nb^nc^n$ is CSL but not CFL, and therefore decidable. Every finite language is regular. A stronger grammar or TM can still recognize any of these lower-family languages.

Study-note section AUT26

> Unary prime length {a^p:p prime} is nonregular: choose prime q above pumping length p; if pump piece has length k≥1, using exponent q+1 gives length q(k+1), composite. Every unary CFL is regular, so this language is not CFL either. It is nevertheless context-sensitive: count the input in bounded workspace and test divisors. The balanced-three-count, prime-length, palindrome and a^nb examples above are all recursive languages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string s∈(V∪Σ)∗;

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 765

Question 73 | AM0766

What is the least expressive listed machine family that recognizes $L=\{a^nb^n \mid n\geq 0\}$? Interpret D as a nondeterministic finite automaton.

A. Finite automaton

B. Pushdown automaton

C. Turing machine

D. Nondeterministic finite automaton

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer B: Pushdown automaton

A PDA pushes one stack symbol for each a, then pops one for each b, accepting exactly when the blocks finish together. The language is nonregular, so neither deterministic nor nondeterministic finite automata suffice. A TM can simulate the matching procedure, but has greater expressive power than required. The least sufficient listed family is the pushdown automaton.

Why each choice is right or wrong

A - Incorrect: Finite automata cannot recognize the nonregular matched-count language.

B - Correct: A PDA stack matches each a with one later b.

C - Incorrect: A TM works but is stronger than necessary.

D - Incorrect: An NFA recognizes only regular languages.

Primary note proof

Study-note section AUT22

> For a^nb^n, push one X per a, switch to a b-reading phase, pop one X per b, and accept when input ends at the bottom marker. Reject a after entering the b phase and b when no X remains. Handle ε explicitly if n=0 is permitted. A stack counts without storing the full input as random-access memory.

Study-note section AUT15

> If L is regular, some $p\geq 1$ satisfies: every $s\in L$ with $|s|\geq p$ has a split $s=xyz$, $|xy|\leq p$, $|y|\geq 1$, and $xy^iz\in L$ for every $i\geq 0$.

Study-note section AUT15

> For $L=\{a^nb^n : n\geq 0\}$, choose a^pb^p. Every allowed $y=a^k$ with $k\geq 1$; pumping $i=0$ leaves fewer a's than b's. The contradiction excludes every DFA.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

R is a set of production rules where each production rule maps a variable to a string $s \in (V \cup \Sigma)*$;

Automata and Complexity Theory module | PDF page 9 | lines 23-23

Pushdown automata accept context-free languages, which include the set of regular languages.

Automata and Complexity Theory module | PDF page 8 | lines 12-12

(1) $|uv| \leq n$ (2) $|v| \geq 1$ (3) for all $i \geq 0$: $uviw \in L$

Automata and Complexity Theory module | PDF page 10 | lines 6-6

Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 766

Question 74 | AM0767

Over $\Sigma=\{a,b\}$, what is the least expressive listed grammar family generating $L=\{ww^R \mid w \in \Sigma^*\}$? Use the usual isolated-start ε convention.

A. Regular grammar

B. Context-free grammar

C. Context-sensitive grammar

D. Unrestricted grammar

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: Context-free grammar

The grammar $S \to aSa \mid bSb \mid \varepsilon$ generates precisely binary even-length palindromes: matching symbols wrap a shorter mirror word until the empty center is reached. This language is context-free and nonregular over the stated two-symbol alphabet. Context-sensitive and unrestricted grammars also suffice but exceed the minimum expressive power required. Fixing the alphabet excludes the unary even-length case, which would be regular.

Why each choice is right or wrong

A - Incorrect: Binary unmarked even palindromes are nonregular.

B - Correct: The wrapping CFG generates exactly this mirror language.

C - Incorrect: Context-sensitive grammars are stronger than necessary.

D - Incorrect: Unrestricted grammars are stronger than necessary.

Primary note proof

Study-note section AUT22

For even palindromes ww^R over {a,b}, push the first half, nondeterministically guess its midpoint, then pop matching symbols. No center marker makes that language context-free but not deterministic context-free. With a marker wcw^R the switch point is visible and a DPDA suffices. Unary even palindromes are merely even-length words and regular. A PDA cannot compare two arbitrary halves in the same forward order: {ww:w∈{a,b}*} is the copy language, not the palindrome language.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

R is a set of production rules where each production rule maps a variable to a string $s \in (V \cup \Sigma)*$;

Automata and Complexity Theory module | PDF page 9 | lines 23-23

Pushdown automata accept context-free languages, which include the set of regular languages.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 8 | lines 12-12

(1) $|uv| \leq n$ (2) $|v| \geq 1$ (3) for all $i \geq 0$: $uviw \in L$

Question sources

Computer Science MCQ practice collection · Question 767

Question 75 | AM0768

Over Σ={a,b}, what is the least expressive listed machine family recognizing $L=\{ww^R \mid w \in \Sigma^*\}$? There is no center marker; D means NFA.

A. Finite automaton

B. Nondeterministic pushdown automaton

C. Turing machine

D. Nondeterministic finite automaton

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: Nondeterministic pushdown automaton

An NPDA pushes the first half, guesses the midpoint and matches the remaining symbols against the stack in reverse order. That recognizes the binary even-palindrome language, which is nonregular. Finite automata and NFAs therefore fail, whereas a TM can recognize it but is stronger than necessary. The unmarked midpoint requires nondeterminism in the PDA family, as distinguished in the note.

Why each choice is right or wrong

A - Incorrect: A finite automaton cannot recognize binary unmarked even palindromes.

B - Correct: A nondeterministic PDA guesses the midpoint and matches using its stack.

C - Incorrect: A TM is sufficient but stronger than necessary.

D - Incorrect: An NFA still recognizes only regular languages.

Primary note proof

Study-note section AUT22

> For even palindromes ww^R over {a,b}, push the first half, nondeterministically guess its midpoint, then pop matching symbols. No center marker makes that language context-free but not deterministic context-free. With a marker wcw^R the switch point is visible and a DPDA suffices. Unary even palindromes are merely even-length words and regular. A PDA cannot compare two arbitrary halves in the same forward order: {ww:w∈{a,b}*} is the copy language, not the palindrome language.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string $s \in (V \cup \Sigma)^*$;

Automata and Complexity Theory module | PDF page 9 | lines 23-23

> Pushdown automata accept context-free languages, which include the set of regular languages.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 8 | lines 12-12

> (1) $|uv| \leq n$ (2) $|v| \geq 1$ (3) for all $i \geq 0$: $uviw \in L$

Question sources

Computer Science MCQ practice collection · Question 768

Question 76 | AM0769

What is the least expressive listed grammar family generating $L=\{0^n1^n2^n \mid n\geq 0\}$? Use the usual isolated-start ε convention.

A. Regular grammar

B. Context-free grammar

C. Context-sensitive grammar

D. Unrestricted grammar

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer C: Context-sensitive grammar

Renaming the symbols in $a^nb^nc^n$ gives $0^n1^n2^n$ without changing its classification. The three matched blocks form a context-sensitive language but not a context-free one. The isolated-start ε convention covers n=0. Regular grammars are weaker still, and unrestricted grammars can generate it but exceed what is needed. Thus the least sufficient listed grammar family is context-sensitive.

Why each choice is right or wrong

A - Incorrect: Regular grammars cannot generate this nonregular language.

B - Incorrect: The language lies outside the context-free family.

C - Correct: The context-sensitive family suffices and is the least listed family that does.

D - Incorrect: Unrestricted grammars suffice but exceed the required expressive power.

Primary note proof

Study-note section AUT25

> A context-sensitive rule $\alpha A\beta\rightarrow\alpha\gamma\beta$ replaces A in context, with γ nonempty. Equivalently, noncontracting grammar presentations preserve/increase total length, with the isolated ε exception. The condition is on the complete sides, not an unrelated comparison $|\beta|\geq|\alpha|$.

Study-note section AUT25

> A linear bounded automaton (LBA) is a TM with O(n) tape cells on length-n input; the standard CSL characterization uses nondeterministic LBAs. For $a^nb^nc^n$ first verify a*b*c* order, repeatedly mark one unmarked a, b and c, and reject missing/excess symbols.

Study-note section AUT25

> Thus every CSL is decidable.

Study-note section AUT24

If L is context-free, some p ensures each sufficiently long $s \in L$ has $s=uvxyz$ with $|vxy| \le p$, $|vy| \ge 1$ and $uv^ixy^iz \in L$ for every $i \ge 0$. The two pieces v and y are pumped together; either may be empty, not both. A non-CFL proof must defeat every allowed decomposition.

Study-note section AUT24

For $s=a^pb^pc^p$, the window vxy cannot contain both an a and a c. Pumping affects at most two blocks and at least one count. Use i=0 or 2: the unaffected third block retains p while another count changes, or symbol order breaks. Hence $a^nb^nc^n$ is not CFL.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

R is a set of production rules where each production rule maps a variable to a string $s \in (V \cup \Sigma)^*$;

Automata and Complexity Theory module | PDF page 10 | lines 6-6

Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 769

Question 77 | AM0770

Which listed machine family can recognize $L=\{0^n1^n2^n \mid n \ge 0\}$? Interpret D as a nondeterministic finite automaton.

A. Finite automaton

B. Pushdown automaton

C. Turing machine

D. Nondeterministic finite automaton

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer C: Turing machine

A Turing machine verifies the 0*1*2* order, repeatedly marks one symbol from each block, and checks that none remains unmatched. Each round marks additional symbols, so the procedure halts. This is the three-equal-count language with renamed symbols, which is not context-free. Consequently neither a finite automaton, an NFA nor a single-stack PDA recognizes it, leaving the TM as the sufficient listed model.

Why each choice is right or wrong

A - Incorrect: Finite automata cannot recognize this nonregular language.

B - Incorrect: A single-stack PDA cannot recognize this non-CFL language.

C - Correct: A TM implements the terminating membership test described in the note.

D - Incorrect: An NFA recognizes only REG and cannot recognize this language.

Primary note proof

Study-note section AUT25

A context-sensitive rule $\alpha A\beta \to \alpha\gamma\beta$ replaces A in context, with γ nonempty. Equivalently, noncontracting grammar presentations preserve/increase total length, with the isolated ε exception. The condition is on the complete sides, not an unrelated comparison $|\beta| \geq |\alpha|$.

Study-note section AUT25

A linear bounded automaton (LBA) is a TM with O(n) tape cells on length-n input; the standard CSL characterization uses nondeterministic LBAs. For $a^n b^n c^n$ first verify a*b*c* order, repeatedly mark one unmarked a, b and c, and reject missing/excess symbols.

Study-note section AUT25

Thus every CSL is decidable.

Study-note section AUT24

If L is context-free, some p ensures each sufficiently long $s \in L$ has $s=uvxyz$ with $|vxy| \leq p$, $|vy| \geq 1$ and $uv^i xy^i z \in L$ for every $i \geq 0$. The two pieces v and y are pumped together; either may be empty, not both. A non-CFL proof must defeat every allowed decomposition.

Study-note section AUT24

For $s=a^p b^p c^p$, the window vxy cannot contain both an a and a c. Pumping affects at most two blocks and at least one count. Use i=0 or 2: the unaffected third block retains p while another count changes, or symbol order breaks. Hence $a^n b^n c^n$ is not CFL.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 11 | lines 7-7

A Turing machine can do everything that a real computer can do.

Automata and Complexity Theory module | PDF page 15 | lines 24-26

A set X is recursive if we have an algorithm to
determine whether a given element belongs to
X or not.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

Fig: The traditional Chomsky hierarchy

Question sources

Computer Science MCQ practice collection · Question 770

Question 78 | AM0771

What is the least expressive listed grammar family generating $L=\{a^n \mid n \text{ is prime}\}$? Use the usual isolated-start ε convention.

A. Regular grammar

B. Context-free grammar

C. Context-sensitive grammar

D. Unrestricted grammar

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: Context-sensitive grammar

Unary prime length is nonregular by the pumping argument. Since every unary context-free language is regular, it is not context-free either. Bounded-workspace counting and divisor testing establish context sensitivity. Unrestricted grammars also suffice but are more expressive than necessary. The least sufficient listed family is therefore context-sensitive, rather than one of the two weaker grammar families.

Why each choice is right or wrong

A - Incorrect: Regular grammars cannot generate this nonregular language.

B - Incorrect: The language lies outside the context-free family.

C - Correct: The context-sensitive family suffices and is the least listed family that does.

D - Incorrect: Unrestricted grammars suffice but exceed the required expressive power.

Primary note proof

Study-note section AUT26

> a^*b^* and $\{a^nb : n \geq 0\}$ are regular. a^nb^n is CFL but not regular. Over a two-symbol alphabet, ww^R is CFL, ww is not CFL; both are decidable. $a^nb^nc^n$ is CSL but not CFL, and therefore decidable. Every finite language is regular. A stronger grammar or TM can still recognize any of these lower-family languages.

Study-note section AUT26

> Unary prime length $\{a^p : p \text{ prime}\}$ is nonregular: choose prime q above pumping length p; if pump piece has length $k \geq 1$, using exponent q+1 gives length q(k+1), composite. Every unary CFL is regular, so this language is not CFL either. It is nevertheless context-sensitive: count the input in bounded workspace and test divisors. The balanced-three-count, prime-length, palindrome and a^nb examples above are all recursive languages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 12-12

> (1) $|uv| \leq n$ (2) $|v| \geq 1$ (3) for all $i \geq 0$: $uviw \in L$

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 15 | lines 24-26

> A set X is recursive if we have an algorithm to
> determine whether a given element belongs to
> X or not.

Question sources

Computer Science MCQ practice collection · Question 771

Question 79 | AM0772

Which listed machine family can recognize $L=\{a^n \mid n \text{ is prime}\}$? Interpret D as a nondeterministic finite automaton.

A. Finite automaton

B. Pushdown automaton

C. Turing machine

D. Nondeterministic finite automaton

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: Turing machine

A Turing machine can count the input and test finitely many divisors to determine whether its length is prime. This procedure halts on every input. The language is nonregular, and every unary CFL is regular, so it is also outside CFL. Finite automata, NFAs and single-stack PDAs therefore fail, while the TM provides the required recognition procedure.

Why each choice is right or wrong

A - Incorrect: Finite automata cannot recognize this nonregular language.

B - Incorrect: A single-stack PDA cannot recognize this non-CFL language.

C - Correct: A TM implements the terminating membership test described in the note.

D - Incorrect: An NFA recognizes only REG and cannot recognize this language.

Primary note proof

Study-note section AUT26

a*b* and {$a^n b$:n≥0} are regular. $a^n b^n$ is CFL but not regular. Over a two-symbol alphabet, ww^R is CFL, ww is not CFL; both are decidable. $a^n b^n c^n$ is CSL but not CFL, and therefore decidable. Every finite language is regular. A stronger grammar or TM can still recognize any of these lower-family languages.

Study-note section AUT26

Unary prime length {a^p:p prime} is nonregular: choose prime q above pumping length p; if pump piece has length k≥1, using exponent q+1 gives length q(k+1), composite. Every unary CFL is regular, so this language is not CFL either. It is nevertheless context-sensitive: count the input in bounded workspace and test divisors. The balanced-three-count, prime-length, palindrome and $a^n b$ examples above are all recursive languages.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 11 | lines 7-7

A Turing machine can do everything that a real computer can do.

Automata and Complexity Theory module | PDF page 15 | lines 24-26

A set X is recursive if we have an algorithm to
determine whether a given element belongs to
X or not.

Automata and Complexity Theory module | PDF page 8 | lines 12-12

(1) |uv| ≤ n (2) |v| ≥ 1 (3) for all i ≥ 0: uviw ∈ L

Question sources

Computer Science MCQ practice collection · Question 772

Question 80 | AM0773

What is the least expressive listed grammar family generating L={$a^n b$ | n≥0}?

A. Regular grammar

B. Context-free grammar

C. Context-sensitive grammar

D. Unrestricted grammar

Hint

Track mandatory symbols and factors that may repeat or disappear. Use short test words to check both directions of each proposed language description.

Answer A: Regular grammar

A right-linear grammar S→aS|b generates any number of a symbols followed by exactly one b, including the word b for n=0. The language is therefore regular. Context-free, context-sensitive and unrestricted grammars can all generate regular languages too, but are more expressive than necessary. The least-family question uniquely selects regular grammar without denying the capabilities of the stronger classes.

Why each choice is right or wrong

A - Correct: The right-linear rule S→aS|b suffices.

B - Incorrect: CFGs work but are stronger than necessary.

C - Incorrect: Context-sensitive grammars work but are stronger than necessary.

D - Incorrect: Unrestricted grammars work but are stronger than necessary.

Primary note proof

Study-note section AUT14

> Right-linear rules have form A→aB, A→a or permitted ε rules; left-linear rules put the single variable at the opposite end, A→Ba. Both consistent orientations generate REG; arbitrarily mixing them can generate nonregular languages, for example S→aA|ε, A→Sb generates $a^n b^n$. A CFG rule A→aBA is legal but not regular-grammar form.

Study-note section AUT14

> Regular languages are closed under union, concatenation, star, reversal, intersection, difference and complement.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 8 | lines 2-2

> The language defined by regular grammar is known as regular language.

Automata and Complexity Theory module | PDF page 10 | lines 6-6

> Fig: The traditional Chomsky hierarchy

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string s∈(V∪Σ)⁎;

Question sources

Computer Science MCQ practice collection · Question 773

Question 81 | AM0774

Which class is defined as the decision languages decidable by a deterministic algorithm in polynomial time?

A. P

B. NP

C. PSPACE

D. EXPTIME

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer A: P

P is defined by deterministic polynomial-time decidability. NP, PSPACE and EXPTIME all contain P, so broad containment would not distinguish the choices. They have different defining bounds: nondeterministic polynomial time, deterministic polynomial workspace and deterministic exponential time, respectively. The explicit deterministic-time definition uniquely identifies P without assuming any unresolved separation among neighboring classes.

Why each choice is right or wrong

A - Correct: P is defined by deterministic polynomial-time decisions.

B - Incorrect: NP is defined by nondeterministic polynomial time or polynomial verification.

C - Incorrect: PSPACE is defined by polynomial workspace.

D - Incorrect: EXPTIME is defined by exponential-time bounds rather than precisely the stated polynomial-time bound.

Primary note proof

Study-note section AUT34

> Complexity classes here contain decision languages. P consists of deterministic polynomial-time decidable languages; NP consists of nondeterministic polynomial-time decidable languages, equivalently polynomially verifiable yes-instances with polynomial-size certificates. A verifier must accept some certificate exactly for yes-inputs; no certificate may falsely establish a no-instance. NP means nondeterministic polynomial, not non-polynomial.

Study-note section AUT34

> P⊆NP because a verifier can ignore its certificate and run the P algorithm. P=NP remains open; NP does not mean definitely hard or definitely outside P.

Study-note section AUT39

> Use L⊆NL⊆P⊆NP⊆PSPACE⊆EXPTIME⊆EXPSPACE. Here L/NL use logarithmic workspace; EXPTIME is ⋃kTIME(2^(n^k)), an upper-bound class that also contains P, not a statement that every member requires exponential time.

Study-note section AUT39

> P versus NP, NP versus PSPACE and PSPACE versus EXPTIME remain unresolved. coNP contains complements of NP languages; P⊆NP∩coNP, and NP=coNP is open.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 17 | lines 27-28

> P problems are set of problems which can be solved in
> polynomial time by deterministic algorithms.

Automata and Complexity Theory module | PDF page 17 | lines 29-30

> NP problems are the problems which can be
> solved in non-deterministic polynomial time.

Question sources

Computer Science MCQ practice collection · Question 774

Question 82 | AM0775

Which class consists of decision problems whose yes-instances have polynomial-size certificates checkable by a deterministic polynomial-time verifier?

A. P

B. NP

C. PSPACE

D. EXPTIME

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer B: NP

NP is defined by polynomially bounded certificates and deterministic polynomial-time verification, equivalently nondeterministic polynomial-time decision. P instead names deterministic polynomial-time decision; its members also admit efficient verification, but P is not the name for the full verifier-defined class. PSPACE bounds deterministic working space by a polynomial, and EXPTIME bounds deterministic time exponentially. These are different resource-based definitions; their inclusion relationships do not make them alternative names for NP.

Why each choice is right or wrong

A - Incorrect: P is defined by deterministic polynomial-time decision, not by the full certificate-verification characterization.

B - Correct: The certificate-and-verifier definition is precisely NP.

C - Incorrect: PSPACE is defined by a polynomial bound on deterministic space.

D - Incorrect: EXPTIME is defined by exponential deterministic time bounds.

Primary note proof

Study-note section AUT34

> Complexity classes here contain decision languages. P consists of deterministic polynomial-time decidable languages; NP consists of nondeterministic polynomial-time decidable languages, equivalently polynomially verifiable yes-instances with polynomial-size certificates. A verifier must accept some certificate exactly for yes-inputs; no certificate may falsely establish a no-instance. NP means nondeterministic polynomial, not non-polynomial.

Study-note section AUT34

> P⊆NP because a verifier can ignore its certificate and run the P algorithm. P=NP remains open; NP does not mean definitely hard or definitely outside P.

Study-note section AUT38

> PSPACE is $\bigcup_k \mathrm{SPACE}(n^k)$, using deterministic polynomial workspace. NP⊆PSPACE because candidate polynomial witnesses can be enumerated and checked one at a time, reusing memory; exponentially many candidates do not require retaining them all.

Study-note section AUT39

> Use L⊆NL⊆P⊆NP⊆PSPACE⊆EXPTIME⊆EXPSPACE. Here L/NL use logarithmic workspace; EXPTIME is $\bigcup_k \mathrm{TIME}(2^{(n^k)})$, an upper-bound class that also contains P, not a statement that every member requires exponential time.

Study-note section AUT39

> P versus NP, NP versus PSPACE and PSPACE versus EXPTIME remain unresolved. coNP contains complements of NP languages; P⊆NP∩coNP, and NP=coNP is open.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 17 | lines 29-30

> NP problems are the problems which can be
> solved in non-deterministic polynomial time.

Automata and Complexity Theory module | PDF page 17 | lines 33-35

> The problem belongs to NP, if it‘s easy to check
> a solution that may have been very tedious to
> find.

Automata and Complexity Theory module | PDF page 17 | lines 27-28

> P problems are set of problems which can be solved in
> polynomial time by deterministic algorithms.

Question sources

Computer Science MCQ practice collection · Question 775

Question 83 | AM0776

Which class is defined as the decision languages decidable using deterministic polynomial workspace?

A. P

B. NP

C. PSPACE

D. EXPTIME

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer C: PSPACE

PSPACE consists of decision languages decidable with deterministic workspace bounded by a polynomial in input length. P and NP use polynomial-time definitions, whereas EXPTIME uses an exponential-time definition and contains PSPACE. Those containments do not change which class the workspace condition defines. Nor is it necessary to assume unresolved strict separations among the classes to identify the definition.

Why each choice is right or wrong

A - Incorrect: P has a deterministic polynomial-time definition.

B - Incorrect: NP has a nondeterministic polynomial-time or certificate-verification definition.

C - Correct: PSPACE has exactly the stated deterministic polynomial-workspace definition.

D - Incorrect: EXPTIME contains PSPACE but is defined by a time bound.

Primary note proof

Study-note section AUT38

PSPACE is $\bigcup_k \mathrm{SPACE}(n^k)$, using deterministic polynomial workspace. NP$\subseteq$PSPACE because candidate polynomial witnesses can be enumerated and checked one at a time, reusing memory; exponentially many candidates do not require retaining them all.

Study-note section AUT34

Complexity classes here contain decision languages. P consists of deterministic polynomial-time decidable languages; NP consists of nondeterministic polynomial-time decidable languages, equivalently polynomially verifiable yes-instances with polynomial-size certificates. A verifier must accept some certificate exactly for yes-inputs; no certificate may falsely establish a no-instance. NP means nondeterministic polynomial, not non-polynomial.

Study-note section AUT34

P⊆NP because a verifier can ignore its certificate and run the P algorithm. P=NP remains open; NP does not mean definitely hard or definitely outside P.

Study-note section AUT39

Use L⊆NL⊆P⊆NP⊆PSPACE⊆EXPTIME⊆EXPSPACE. Here L/NL use logarithmic workspace; EXPTIME is ⋃kTIME(2^(n^k)), an upper-bound class that also contains P, not a statement that every member requires exponential time.

Study-note section AUT39

P versus NP, NP versus PSPACE and PSPACE versus EXPTIME remain unresolved. coNP contains complements of NP languages; P⊆NP∩coNP, and NP=coNP is open.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 16 | lines 4-5

Space Complexity: Determine the approximate memory required to solve a problem of size n.

Automata and Complexity Theory module | PDF page 17 | lines 27-28

P problems are set of problems which can be solved in polynomial time by deterministic algorithms.

Automata and Complexity Theory module | PDF page 17 | lines 29-30

NP problems are the problems which can be solved in non-deterministic polynomial time.

Question sources

Computer Science MCQ practice collection · Question 776

Question 84 | AM0777

Which concept is used to show that one problem can be reduced to another problem in order to prove hardness or completeness?

A. Reductions

B. Hierarchy

C. Completeness

D. Complexity classes

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer A: Reductions.

AUT35 defines a polynomial reduction as a computable transformation preserving yes/no membership and explains how reducing a known hard problem to a new target proves target hardness. AUT32 supplies the computable version for undecidability. Thus reductions name the proof concept itself, while hierarchy, completeness and classes describe relationships or resulting properties rather than the transformation method.

Why each choice is right or wrong

A - Correct: A reduction transforms instances while preserving membership, enabling hardness or completeness arguments with the stated resource bound.

B - Incorrect: A hierarchy organizes classes; it does not itself supply the instance transformation used in a hardness proof.

C - Incorrect: Completeness is the property proved using membership and hardness, not the reduction method itself.

D - Incorrect: Complexity classes group problems by resources; reductions establish the required relationships between individual problems.

Primary note proof

Study-note section AUT35

> $A \leq_p B$ means f is computable in polynomial time and $x \in A$ iff $f(x) \in B$. Solving B then solves A, so B is at least as hard under this reduction. If $B \in P$ then $A \in P$; if A is NP-hard then B is NP-hard. $A \in P$ does not force $B \in P$, and the reduction alone does not place both problems in the same class. Polynomial reductions compose.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 15 | lines 33-35

> Algorithm analysis is an important part of a broader computational complexity theory, which provides theoretical estimates for the resources needed by any algorithm which solves a given computational problem.

Automata and Complexity Theory module | PDF page 17 | lines 29-30

> NP problems are the problems which can be solved in non-deterministic polynomial time.

Question sources

Computer Science MCQ practice collection · Question 777

Question 85 | AM0778

Which listed term denotes decision problems that belong to NP and are at least as hard as every NP problem under polynomial-time many-one reductions?

A. P

B. NP-complete

C. PSPACE

D. EXPTIME

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer B: NP-complete

NP-completeness combines two conditions: the target problem belongs to NP, and every NP language reduces to it in polynomial time. That gives a precise meaning to hardest within NP under the chosen reductions. P, PSPACE and EXPTIME instead name resource-bounded language classes and do not by themselves assert that each member is hard for NP or belongs to NP.

Why each choice is right or wrong

A - Incorrect: P names a resource class, not the membership-plus-hardness condition.

B - Correct: NP-complete states both NP membership and hardness for all NP languages.

C - Incorrect: PSPACE membership alone does not give the specified NP completeness.

D - Incorrect: EXPTIME membership alone does not give the specified NP completeness.

Primary note proof

Study-note section AUT36

> X is NP-hard if every A∈NP satisfies A≤pX. It need not belong to NP and can even be undecidable. X is NP-complete if X∈NP and X is NP-hard. NP-complete languages lie inside NP and can be decided by finite exhaustive certificate search in exponential time. They are not known to require exponential time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 17 | lines 27-28

> P problems are set of problems which can be solved in
> polynomial time by deterministic algorithms.

Automata and Complexity Theory module | PDF page 17 | lines 29-30

> NP problems are the problems which can be
> solved in non-deterministic polynomial time.

Automata and Complexity Theory module | PDF page 15 | lines 33-35

Algorithm analysis is an important part of a broader computational complexity theory, which provides theoretical estimates for the resources needed by any algorithm which solves a given computational problem.

Question sources

Computer Science MCQ practice collection · Question 778

Question 86 | CS0247

Let a Grammar G = ({S}, {a,b}, S, P), where P: S -> aSb, S -> lambda. Which one of the following is NOT an element of the language L generated from G?

A. aabb

B. ab

C. lambda (empty string)

D. a

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer D: a

After n applications of S→aSb and a final S→λ the word is a^n b^n. Induction preserves equal counts in block order, including n=0. a has one a and no b and cannot be generated.

Why each choice is right or wrong

A - Incorrect: S⇒aSb⇒aaSbb⇒aabb.

B - Incorrect: S⇒aSb⇒ab.

C - Incorrect: S⇒λ directly.

D - Correct: Unequal counts violate the invariant of every terminal derivation.

Primary note proof

Study-note section AUT17

> Use a recursive rule to enforce matched structure and a base rule to stop. S→aSb|ε generates exactly $a^n b^n$ for n≥0: n recursive steps add n a's on the left and n b's on the right. Derive aabb by S⇒aSb⇒aaSbb⇒aabb. For n≥1 replace ε by base ab.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 8 | lines 27-28

> R is a set of production rules where each production rule maps a variable to a string s∈(V∪Σ)*;

Automata and Complexity Theory module | PDF page 2 | lines 24-25

> A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Question sources

Computer Science Exit Exam practice collection · Question 247

Question 87 | CS0249

Given input Alphabet sigma = {a, b, c, d} and empty string lambda. Which one of the following is true about sigma*?

A. sigma* = sigma+

B. sigma* = sigma+ union {lambda}

C. sigma* = sigma^1 union sigma^2 union sigma^3 union sigma+

D. sigma* = {lambda}

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer B: sigma* = sigma+ union {lambda}

Star allows every finite number of alphabet symbols, including zero. Positive closure allows all positive numbers. Their union with the singleton zero-length word gives exactly star. In C, unioning positive powers with Σ^+ still gives only Σ^+, missing λ.

Why each choice is right or wrong

A - Incorrect: Σ^+ omits λ, so it is smaller than Σ^*.

B - Correct: Partitions all words by zero versus positive length.

C - Incorrect: All terms already have positive length; λ is absent.

D - Incorrect: Omits every nonempty word such as a.

Primary note proof

Study-note section AUT03

> Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, $L^*=\bigcup n\geq 0 L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 5 | lines 25-25

means one or more occurrence of x. It can generate {x, xx, xxx, xxxx,

Automata and Complexity Theory module | PDF page 2 | lines 24-25

A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Question sources

Computer Science Exit Exam practice collection · Question 249

Question 88 | CS0250

Given input Alphabet sigma = {a, b, c, d} and empty string lambda. Which one of the following is equivalent to sigma+?

A. sigma

B. sigma+

C. {lambda}

D. sigma*

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer B: sigma+

Taken literally, B is an identity: $\Sigma+$ equals itself. Since the alphabet is nonempty and contains no empty symbol, Σ contains only length-one words, Σ^* additionally contains λ, and $\{\lambda\}$ contains no positive-length word. The identical expressions make B a definite answer as printed.

Why each choice is right or wrong

A - Incorrect: aa is in $\Sigma+$ but not in Σ.

B - Correct: Identical expression on both sides.

C - Incorrect: λ has length zero and is not a positive repetition.

D - Incorrect: Σ^* includes λ, unlike $\Sigma+$.

Primary note proof

Study-note section AUT03

Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

For an arbitrary language L, $L^*=\bigcup n\geq 0L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set;

> intersection requires the same word in both, not concatenation. Complement is Σ*−L and needs a fixed alphabet.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Automata and Complexity Theory module | PDF page 5 | lines 25-25

> means one or more occurrence of x. It can generate {x, xx, xxx, xxxx,

Automata and Complexity Theory module | PDF page 2 | lines 24-25

> A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Question sources

Computer Science Exit Exam practice collection · Question 250

Question 89 | CS0251

If Z=caddba, what is $|Z|^2$, the square of the numeric length of Z?

A. 3

B. 4

C. 2

D. 36

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer D: 36

String length counts every symbol occurrence, including repetitions. The word caddba has six positions, so $|Z|=6$ and $|Z|^2=6^2=36$. Counting distinct symbols would give four, which is not string length. Concatenating two copies gives length twelve, but that computes $|Z^2|$ instead; the revised stem expressly preserves the original placement of the exponent outside the bars.

Why each choice is right or wrong

A - Incorrect: Three is neither the length nor its square.

B - Incorrect: Four counts distinct letters rather than the squared length.

C - Incorrect: Two is the exponent, not the squared length.

D - Correct: Six positions squared gives 36.

Primary note proof

Study-note section AUT01

> An alphabet Σ is a finite set of symbols. A word/string is a finite ordered sequence of symbols from Σ; a language is a set of words, L⊆Σ*. An empty word ε, also written λ, has length zero. The empty language ∅ contains no words; {ε} contains one.

Study-note section AUT01

> Length counts occurrences, not distinct symbols: |caddba|=6, so $|caddba|^2$=36.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 2 | lines 26-26

> The number of symbols in a string w is called the length of a string. It is denoted by |w|.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

> In a regular expression, x* means zero or more occurrence of x.

Question sources

Computer Science Exit Exam practice collection · Question 251

Question 90 | CS0252

What will be the value of Z^R (reverse) if string Z = caddba?

A. abddac

B. cad

C. abd

D. caddba

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer A: abddac

Reversal reads the characters in the opposite order without dropping any: c a d d b a becomes a b d d a c.

Why each choice is right or wrong

A - Correct: Produces every symbol in reverse order: abddac.

B - Incorrect: Only a three-symbol prefix; does not reverse the full word.

C - Incorrect: Only the first three reversed symbols; drops the remaining characters.

D - Incorrect: Original word differs from its reverse at the first symbol.

Primary note proof

Study-note section AUT02

> Concatenation xy writes x followed by y: cddc·ababb=cddcababb; order generally matters. εx=xε=x. Define $x^0=\varepsilon$ and x^n as n copies for n≥1. Reverse x^R reads symbols backward: caddba reverses to abddac; $(xy)^R=y^Rx^R$ and $(x^R)^R=x$.

Study-note section AUT02

> For languages, LM={xy:x∈L,y∈M}, $L^0=\{\varepsilon\}$, and $L^{n+1}=L^nL$. Thus {e,fgh}{c,d}={ec,ed,fghc,fghd}; fghe is absent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 2 | lines 21-21

> It is a finite collection of symbols from the alphabet. The string is denoted by w.

Automata and Complexity Theory module | PDF page 2 | lines 26-26

> The number of symbols in a string w is called the length of a string. It is denoted by |w|.

Question sources

Computer Science Exit Exam practice collection · Question 252

Question 91 | CS0253

If we have two strings Z = cddc and W = ababb, then the concatenation ZW will be?

A. ZW = cddcababb

B. ZW = cdab

C. ZW = ababbcddc

D. ZW = ca

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer A: ZW = cddcababb.

AUT02 defines xy as all of x followed by all of y, preserving order and every symbol. Appending ababb to cddc yields cddcababb, of length 4+5=9. Reversing the operands produces a different string; selecting prefixes or first letters discards symbols and is not concatenation.

Why each choice is right or wrong

A - Correct: This preserves cddc followed by ababb, giving the required nine-symbol concatenation.

B - Incorrect: cdab discards symbols from both operands instead of retaining both complete strings.

C - Incorrect: ababbcddc is WZ, reversing the specified operand order.

D - Incorrect: ca retains only initial symbols and is not the concatenation of the full strings.

Primary note proof

Study-note section AUT02

> Concatenation xy writes x followed by y: cddc·ababb=cddcababb; order generally matters. εx=xε=x. Define $x^0=\varepsilon$ and x^n as n copies for n≥1. Reverse x^R reads symbols backward: caddba reverses to abddac; $(xy)^R=y^Rx^R$ and $(x^R)^R=x$.

Study-note section AUT02

> For languages, LM={xy:x∈L,y∈M}, $L^0=\{\varepsilon\}$, and $L^{n+1}=L^nL$. Thus {e,fgh}{c,d}={ec,ed,fghc,fghd}; fghe is absent.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 2 | lines 21-21

> It is a finite collection of symbols from the alphabet. The string is denoted by w.

Automata and Complexity Theory module | PDF page 2 | lines 26-26

> The number of symbols in a string w is called the length of a string. It is denoted by |w|.

Question sources

Computer Science Exit Exam practice collection · Question 253

Question 92 | CS0254

Given input alphabet sigma = {a, b, c, d} and empty string lambda. Which one of the following is FALSE about lambda?

A. lambda*ab = ab

B. For any string Z, Z^0 = lambda

C. lambda is an element of sigma*

D. lambda = 0

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer D: The empty word is not numeric zero.

AUT01 distinguishes the empty word λ from the numeric symbol 0. AUT02 gives λx=x and Z^0=λ, while AUT03 includes the empty word in Σ*. Thus the first three claims express the standard empty-word identities under the displayed concatenation notation, and D is the false equality.

Why each choice is right or wrong

A - Incorrect: Reading the displayed star as concatenation, the empty word followed by ab is ab.

B - Incorrect: Every string's zeroth concatenation power is the empty word.

C - Incorrect: Kleene closure of the alphabet includes its zero-length word λ.

D - Correct: This equality is false: λ is a zero-length word, not the numeric symbol 0.

Primary note proof

Study-note section AUT01

> An alphabet Σ is a finite set of symbols. A word/string is a finite ordered sequence of symbols from Σ; a language is a set of words, L⊆Σ*. An empty word ε, also written λ, has length zero. The empty language ∅ contains no words; {ε} contains one.

Study-note section AUT01

> Length counts occurrences, not distinct symbols: |caddba|=6, so $|caddba|^2$=36.

Study-note section AUT02

> Concatenation xy writes x followed by y: cddc·ababb=cddcababb; order generally matters. εx=xε=x. Define x^0=ε and x^n as n copies for n≥1. Reverse x^R reads symbols backward: caddba reverses to abddac; $(xy)^R$=y^Rx^R and $(x^R)^R$=x.

Study-note section AUT02

> For languages, LM={xy:x∈L,y∈M}, L^0={ε}, and L^{n+1}=L^nL. Thus {e,fgh}{c,d}={ec,ed,fghc,fghd}; fghe is absent.

Study-note section AUT03

> $Σ^n$ contains words of exactly n symbols; for |Σ|=k, $|Σ^n|$=k^n, including $Σ^0$={ε}. Σ*=∪n≥0$Σ^n$ and $Σ^+$=∪n≥1$Σ^n$. Over a nonempty alphabet, $Σ^+$ excludes ε and Σ*=$Σ^+$∪{ε}. $Σ^2$ over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, L*=∪n≥0L^n and L^+=LL*. Unlike alphabet positive closure, L^+ can contain ε when ε∈L. ∅*={ε}, $∅^+$=∅, and {ε}*={ε}. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ*−L and needs a fixed alphabet.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 2 | lines 24-25

A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Automata and Complexity Theory module | PDF page 5 | lines 22-22

In a regular expression, x* means zero or more occurrence of x.

Question sources

Computer Science Exit Exam practice collection · Question 254

Question 93 | CS0255

Which one of the following is FALSE about any language L?

A. L^3 = L^1 * L^1 * L^1

B. L^0 = lambda

C. L* = L^0 union L^1 union L^2 union L^3 ...

D. L+ = L^1 union L^2 union L^3 ...

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer B: L^0 is {λ}, not the word λ itself.

AUT02 defines $L^0=\{\varepsilon\}$; AUT01 equates λ with ε. Thus the literal equality $L^0=\lambda$ confuses a set with its sole word and is false unless braces are explicitly understood. Reading the binary stars in A as the displayed concatenation separators, A is the third-power definition; C and D are the unions defining star and positive closure in AUT03.

Why each choice is right or wrong

A - Incorrect: With the displayed binary stars denoting concatenation, three copies of L form L^3, so this is true.

B - Correct: This is the false statement: the zeroth power is the singleton language {λ}, not the word λ.

C - Incorrect: The union of all nonnegative language powers is the definition of Kleene closure, so this is true.

D - Incorrect: The union of positive language powers is positive closure, even if L itself contains ε.

Primary note proof

Study-note section AUT03

Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

For an arbitrary language L, $L^*=\cup n\geq 0 L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Study-note section AUT02

Concatenation xy writes x followed by y: cddc·ababb=cddcababb; order generally matters. εx=xε=x. Define $x^0=\varepsilon$ and x^n as n copies for n≥1. Reverse x^R reads symbols backward: caddba reverses to abddac; $(xy)^R=y^Rx^R$ and $(x^R)^R=x$.

Study-note section AUT02

For languages, LM={xy:x∈L,y∈M}, $L^0=\{\varepsilon\}$, and $L^{n+1}=L^nL$. Thus {e,fgh}{c,d}={ec,ed,fghc,fghd}; fghe is absent.

Study-note section AUT01

An alphabet Σ is a finite set of symbols. A word/string is a finite ordered sequence of symbols from Σ; a language is a set of words, $L\subseteq\Sigma^*$. An empty word ε, also written λ, has length zero. The empty language ∅ contains no words; {ε} contains one.

Study-note section AUT01

Length counts occurrences, not distinct symbols: $|caddba|=6$, so $|caddba|^2=36$.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 6 | lines 4-4

Kleen closure: If L is a regular language then its Kleen closure L1* will also be a regular language.

Automata and Complexity Theory module | PDF page 2 | lines 24-25

A string with zero occurrences of symbols is known as an empty string. It is represented by ε.

Question sources

Computer Science Exit Exam practice collection · Question 255

Question 94 | CS0256

Given input alphabet sigma = {a, b, c, d}, which one of the following is an element of set sigma^2?

A. b

B. c

C. ac

D. aabbccdd

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer C: ac.

AUT03 defines Σ^n as the words containing exactly n alphabet symbols. The given alphabet includes a and c, so ac has two permitted symbols and belongs to Σ^2. Single-symbol choices have length one; aabbccdd has length eight. Membership depends on length and symbol validity, not on including all alphabet symbols.

Why each choice is right or wrong

A - Incorrect: b uses an allowed symbol but has length one rather than the required two.

B - Incorrect: c is a one-symbol word and therefore belongs to Σ^1, not Σ^2.

C - Correct: ac has exactly two symbols, both drawn from the given alphabet.

D - Incorrect: aabbccdd has eight symbols; using each symbol twice does not make it a length-two word.

Primary note proof

Study-note section AUT03

> Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, $L^*=\bigcup n\geq 0L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 2 | lines 15-15

> An alphabet finite set of non-empty symbols. It is denoted by ∑ (sigma).

Automata and Complexity Theory module | PDF page 2 | lines 26-26

> The number of symbols in a string w is called the length of a string. It is denoted by |w|.

Automata and Complexity Theory module | PDF page 2 | lines 21-21

> It is a finite collection of symbols from the alphabet. The string is denoted by w.

Question sources

Computer Science Exit Exam practice collection · Question 256

Question 95 | CS0260

In a Turing Machine, the transition a -> b, L means:

A. Read a, write b and move left

B. Delete a, read b and move left

C. Write a, read b and move left

D. Replace b by a and move left

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Read a, write b and move left

The TM transition reads the scanned a, replaces it with b, and moves the head one cell left. This is the transition ordering (read symbol, write symbol, movement), with next state specified by the destination vertex.

Why each choice is right or wrong

A - Correct: Matches the scanned symbol, replacement, and L direction.

B - Incorrect: b is written rather than read; no distinct delete-then-read instruction is specified.

C - Incorrect: Swaps read and write roles.

D - Incorrect: Replaces the wrong symbol in the wrong direction.

Primary note proof

Study-note section AUT27

> δ(q,a)=(p,b,L) means read a in q, write b, move one cell left and enter p. Accept/reject halting states have no further moves under this convention.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Automata and Complexity Theory module | PDF page 11 | lines 20-22

> δ is the transition function, mapping (q, x) onto (q', y, D) where D denotes the direction of movement of R/W head: D = L or R according as the movement is to the left or right.

Automata and Complexity Theory module | PDF page 11 | lines 9-9

> It has a tape head that can read and write symbols and move around on the tape.

Automata and Complexity Theory module | PDF page 14 | lines 8-8

> As a result the symbol is β written in the cell under the R/W head.

Question sources

Computer Science Exit Exam practice collection · Question 260

Question 96 | NS0161

What is the difference between a DFA and an NFA? Are they equally powerful?

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer: DFA has one successor per state/symbol; NFA permits sets. They recognize the same languages.

AUT09 permits an NFA to have zero, one or several next states and states that DFA and NFA recognize exactly REG. Subset construction simulates the simultaneous NFA possibilities by DFA states.

Primary note proof

Study-note section AUT06

> A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0 \in Q$, accepting set $F \subseteq Q$ and total $\delta: Q \times \Sigma \to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition.

Study-note section AUT09

> A nondeterministic finite automaton (NFA) permits zero, one or many successors: $\delta: Q \times \Sigma \to P(Q)$.

Study-note section AUT09

> An NFA is not more expressive than a DFA: both recognize exactly REG. Nondeterminism can make a description smaller.

Study-note section AUT10

> ε-closure(T) contains T and every state reachable through ε moves. Start the DFA at ε-closure($\{q_0\}$); for each current subset T and symbol a compute ε-closure($\bigcup_{q \in T} \delta(q,a)$). A subset accepts iff it intersects F.

Question sources

Automata & Complexity Theory review questions · Question 1

Question 97 | NS0162

What is the pumping lemma used for? Demonstrate with an example.

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer: The regular pumping lemma can disprove regularity, for example for $a^n b^n$.

AUT15 chooses $a^p b^p$ after assuming a pumping length p. Every legal nonempty y in the first p symbols contains only a, so pumping it out changes the a count without changing b. That contradicts regularity.

Primary note proof

Study-note section AUT15

> If L is regular, some $p \geq 1$ satisfies: every $s \in L$ with $|s| \geq p$ has a split $s=xyz$, $|xy| \leq p$, $|y| \geq 1$, and $xy^i z \in L$ for every $i \geq 0$.

Study-note section AUT15

> For $L=\{a^n b^n : n \geq 0\}$, choose $a^p b^p$. Every allowed $y=a^k$ with $k \geq 1$; pumping $i=0$ leaves fewer a's than b's. The contradiction excludes every DFA.

Question sources

Automata & Complexity Theory review questions · Question 2

Question 98 | NS0163

What is the difference between P and NP? Why does it matter?

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer: P is deterministic polynomial-time decision; NP admits polynomial certificates. Equality remains open.

AUT34 defines P by deterministic polynomial running time and NP by nondeterministic polynomial time or polynomial-size verifiable certificates. P is included in NP, and equality is unresolved. This matters for converting efficiently checked witnesses into efficiently found decisions.

Primary note proof

Study-note section AUT34

> Complexity classes here contain decision languages. P consists of deterministic polynomial-time decidable languages; NP consists of nondeterministic polynomial-time decidable languages, equivalently polynomially verifiable yes-instances with polynomial-size certificates. A verifier must accept some certificate exactly for yes-inputs; no certificate may falsely establish a no-instance. NP means nondeterministic polynomial, not non-polynomial.

Study-note section AUT34

> P⊆NP because a verifier can ignore its certificate and run the P algorithm. P=NP remains open; NP does not mean definitely hard or definitely outside P. Finding versus verifying is an intuition for witnesses, not a rule that all NP instances need exponential search.

Study-note section AUT36

> X is NP-hard if every A∈NP satisfies A≤pX. It need not belong to NP and can even be undecidable. X is NP-complete if X∈NP and X is NP-hard.

Question sources

Automata & Complexity Theory review questions · Question 3

Question 99 | NS0164

Why is the halting problem undecidable? Explain the contradiction.

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer: A proposed halting decider yields a program contradicting its own prediction.

AUT30 constructs D(x) to loop when H(x,x) predicts halting and to halt otherwise. On D's own description, either prediction is contradicted. Therefore no general halting decider exists, although simulation recognizes halting instances.

Primary note proof

Study-note section AUT30

> HALT={⟨M,w⟩:M halts on w} is recognizable: simulate and accept upon any halt. Suppose a decider H determines halting. Construct D(x) that loops if H(x,x) says halt, otherwise halts. Running D on its own description contradicts H in both cases. Thus HALT is undecidable and witnesses RE beyond DECIDABLE. Its complement is not RE, or both recognizers would yield a decider.

Question sources

Automata & Complexity Theory review questions · Question 4

Question 100 | NS0165

What is the Chomsky hierarchy and why are the containments proper?

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer: REG⊊CFL⊊CSL⊊RE by increasing generative power; separating examples witness proper inclusions.

AUT05 supplies the proper family hierarchy. Equal pairs separate CFL from REG; equal triples separate CSL from CFL. CSLs are decidable by AUT25, while HALT is RE but undecidable under AUT30, separating CSL from RE.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Study-note section AUT26

> a*b* and $\{a^n b : n \geq 0\}$ are regular. $a^n b^n$ is CFL but not regular. Over a two-symbol alphabet, ww^R is CFL, ww is not CFL; both are decidable. $a^n b^n c^n$ is CSL but not CFL, and therefore decidable. Every finite language is regular. A stronger grammar or TM can still recognize any of these lower-family languages.

Study-note section AUT30

> Thus HALT is undecidable and witnesses RE beyond DECIDABLE.

Question sources

Automata & Complexity Theory review questions · Question 5

Question 101 | NS0166

Which of the following correctly orders automata from LEAST to MOST powerful?

A. NFA < DFA < PDA < TM

B. PDA < NFA < DFA < TM

C. DFA = NFA < PDA < LBA < TM

D. TM < LBA < PDA < DFA

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: DFA=NFA < PDA < LBA < TM in language-recognition power.

AUT09 equates DFA and NFA power. AUT05 assigns finite automata, nondeterministic PDAs, nondeterministic LBAs and Turing recognizers to successively stronger language families. That gives C under the standard nondeterministic PDA/LBA family convention. The comparison is expressiveness, not a statement that stronger models always run faster.

Why each choice is right or wrong

A - Incorrect: NFA and DFA have equal power rather than the strict order shown.

B - Incorrect: PDAs recognize more than regular languages, so placing PDA below finite automata reverses the hierarchy.

C - Correct: This gives equal finite-automaton power followed by the standard strictly increasing PDA/LBA/TM families.

D - Incorrect: This reverses the standard language-recognition power hierarchy.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Question sources

Automata & Complexity Theory review questions · Question 1

Question 102 | NS0167

Complexity theory primarily studies:

A. How to design finite automata

B. Resources (time and space) required to solve computational problems

C. How to convert NFA to DFA

D. The syntax of formal grammars

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer B: Time and space resources for computational problems.

AUT33 defines time as computation steps and space as memory cells, and AUT34 groups decision problems by polynomial resource bounds. Complexity studies these resource requirements as a function of encoded input size. Designing automata or transforming grammars may be objects of study, but neither is the general defining purpose.

Why each choice is right or wrong

A - Incorrect: Finite-automaton design is one specific task rather than the general study of computational resources.

B - Correct: Complexity theory classifies time and space requirements relative to input size.

C - Incorrect: NFA determinization is a particular construction, not the broad subject of complexity theory.

D - Incorrect: Grammar syntax concerns formal language description rather than resource requirements in general.

Primary note proof

Study-note section AUT-S01

> Computability theory asks whether an effective algorithm can solve a problem at all. For a decision problem, a decision procedure must terminate with the correct yes-or-no result on every input.

Study-note section AUT-S01

> Computational complexity theory studies the time and memory resources required to solve problems as functions of encoded input size. Automata theory studies abstract machine models and the languages they recognize; these subjects overlap but ask different questions.

Question sources

Automata & Complexity Theory review questions · Question 2

Question 103 | NS0168

Given $\Sigma = \{a, b\}$, which of the following is in Σ^2 but NOT in Σ^1?

A. a

B. b

C. ε

D. ab

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer D: ab.

AUT03 defines Σ^n by exact word length. Over {a,b}, ab contains two permitted symbols and is not length one. The single letters a and b belong to Σ^1; ε has length zero. Thus only ab satisfies both membership in Σ^2 and exclusion from Σ^1.

Why each choice is right or wrong

A - Incorrect: a has length one and belongs to Σ^1 rather than Σ^2.

B - Incorrect: b has length one, violating the required exact length two.

C - Incorrect: Epsilon has length zero and is not in Σ^2.

D - Correct: ab has two permitted symbols and is not a one-symbol word.

Primary note proof

Study-note section AUT03

> Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, $L^*=\bigcup n\geq 0 L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Question sources

Automata & Complexity Theory review questions · Question 3

Question 104 | NS0169

The set Σ^* for $\Sigma = \{0, 1\}$ includes which of the following?

A. ε (empty string)

B. All binary strings of length ≥ 1 only

C. Only finite-length strings starting with 1

D. All strings NOT over {0,1}

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer A: The empty string is included.

AUT03 defines Σ^* as the union of all nonnegative powers, including $\Sigma^0=\{\varepsilon\}$. Therefore the empty word belongs to the binary closure. Positive-length-only and starting-with-one descriptions exclude valid members, and strings using symbols outside {0,1} are outside this universe. The wording of B must not erase its restrictive only.

Why each choice is right or wrong

A - Correct: The zero power contributes epsilon to the Kleene closure.

B - Incorrect: The word only excludes epsilon, so this is not a complete description of Σ^*.

C - Incorrect: Binary strings starting with 0 and epsilon also belong, so the starting-with-1 restriction is false.

D - Incorrect: Σ^* consists of words over the specified binary alphabet, not words outside it.

Primary note proof

Study-note section AUT03

> Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, $L^*=\bigcup n\geq 0 L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Question sources

Automata & Complexity Theory review questions · Question 4

Question 105 | NS0170

What is the difference between Σ^* and Σ^+?

A. Σ^* excludes strings of length 2 or more

B. Σ^+ includes the empty string but Σ^* does not

C. Σ^* includes the empty string (ε) but Σ^+ does not

D. They are identical

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer C: Σ^* includes ε; Σ^+ excludes it.

AUT03 explicitly gives $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$ for a nonempty alphabet. Both operations allow arbitrarily long finite words, but only star includes length zero. This statement concerns alphabet closure; the note separately cautions that positive closure of an arbitrary language containing epsilon can itself include epsilon.

Why each choice is right or wrong

A - Incorrect: Σ^* contains words of every finite length, including length two and above.

B - Incorrect: This reverses which alphabet closure contains the empty word.

C - Correct: Star includes the zero-length word and positive alphabet closure omits it.

D - Incorrect: They differ at least by epsilon for the nonempty alphabet convention.

Primary note proof

Study-note section AUT03

> Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, $L^*=\bigcup n\geq 0 L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Question sources

Automata & Complexity Theory review questions · Question 5

Question 106 | NS0171

In the five-tuple $M = (Q, \Sigma, \delta, q_0, F)$ of an FSM, which component determines how the machine moves from state to state?

A. Q - the set of states

B. δ - the transition function

C. F - the set of accepting states

D. q_0 - the initial state

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer B: δ, the transition function.

AUT06 defines $\delta:Q\times\Sigma\to Q$, mapping a current state and input symbol to the next state. That is precisely the movement rule asked for. Q lists available states, F marks acceptance and q_0 fixes the starting state; none of those alone specifies the transitions between states.

Why each choice is right or wrong

A - Incorrect: Q supplies state names but not the next-state rule for input symbols.

B - Correct: Delta maps the current state and input symbol to the next state.

C - Incorrect: F identifies acceptance rather than transitions between states.

D - Incorrect: The initial state specifies where processing starts, not how each symbol changes state.

Primary note proof

Study-note section AUT06

> A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0\in Q$, accepting set $F\subseteq Q$ and total $\delta:Q\times\Sigma\to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition. F may be empty, recognizing $\varnothing$.

Study-note section AUT06

> Start at q_0 and consume the entire input. Accept iff the ending state is in F; visiting an accepting state earlier is insufficient. ε is accepted exactly when $q_0\in F$. A nonaccepting sink loops on every symbol and cannot reach acceptance; it still finishes after finite input. Finite automata have finite memory encoded by states, not no state or no ability to remember any history.

Question sources

Automata & Complexity Theory review questions · Question 6

Question 107 | NS0172

What is the defining characteristic that makes an automaton 'deterministic'?

A. It has exactly one accepting state

B. It processes input from right to left

C. It never enters a dead state

D. For every (state, input) pair, there is exactly ONE next state

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer D: One next state for every state/input-symbol pair.

AUT06 explicitly defines exactly one successor per state/symbol pair. Determinism does not constrain the number of final states, input-reading direction or whether a rejecting sink is reachable. The uniqueness concerns each transition choice, not the number of states or transitions in the entire automaton.

Why each choice is right or wrong

A - Incorrect: A DFA may have zero, one or several accepting states.

B - Incorrect: Right-to-left processing is not the definition of determinism.

C - Incorrect: Deterministic automata may legitimately have reachable rejecting sinks.

D - Correct: The next state is uniquely determined by the current state and current input symbol.

Primary note proof

Study-note section AUT06

> A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0 \in Q$, accepting set $F \subseteq Q$ and total $\delta: Q \times \Sigma \to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition. F may be empty, recognizing $\varnothing$.

Study-note section AUT06

> Start at q_0 and consume the entire input. Accept iff the ending state is in F; visiting an accepting state earlier is insufficient. ε is accepted exactly when $q_0 \in F$. A nonaccepting sink loops on every symbol and cannot reach acceptance; it still finishes after finite input. Finite automata have finite memory encoded by states, not no state or no ability to remember any history.

Question sources

Automata & Complexity Theory review questions · Question 7

Question 108 | NS0173

Over the alphabet Σ={a,b,c}, what is the minimum number of states in a complete DFA accepting exactly the strings that start with abc?

A. 3

B. 5

C. 4

D. 6

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer B: 5

Before completing abc, the DFA must distinguish no matched symbol, a matched, and ab matched. It then needs an accepting state for all continuations after abc and a rejecting sink for any earlier mismatch. These five distinct situations require five states in a complete DFA. Including c in the alphabet makes the intended prefix language nonempty and the standard prefix-state construction applicable.

Why each choice is right or wrong

A - Incorrect: Three omits the accepting continuation and rejecting-mismatch situations.

B - Correct: Three incomplete-prefix stages plus accepting continuation and sink give five.

C - Incorrect: Four omits a necessary distinct situation in the complete DFA.

D - Incorrect: Six exceeds the five-state construction and is not minimal.

Primary note proof

Study-note section AUT08

> For a fixed valid word w of length n, a DFA for strings ending in w needs n+1 states representing the longest consumed suffix that is a prefix of w. Overlapping prefixes determine fallback transitions. Distinct prefixes are distinguishable by suitable continuations, proving the lower bound; do not add an unnecessary rejecting sink.

Study-note section AUT08

> For strings starting with w, a complete DFA ordinarily has n matched-prefix stages, an accepting continuation state and a rejecting sink: n+2 when n≥1 and the alphabet allows mismatches.

Question sources

Automata & Complexity Theory review questions · Question 8

Question 109 | NS0174

In a DFA, what happens after reaching a nonaccepting trap state while processing a finite input?

A. It immediately accepts the input

B. It restarts processing from the initial state

C. It stays there for every remaining input symbol and rejects after the input ends.

D. It transitions to the initial state on the next input

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer C: It stays there for every remaining input symbol and rejects after the input ends.

A nonaccepting trap is a sink: every subsequent input symbol leaves the machine in that same state. It therefore cannot recover an accepting state. Once the finite remaining input is consumed, the DFA finishes and rejects. A self-loop transition does not imply an infinite computation after the input ends, nor does it restart processing or transfer to the start state.

Why each choice is right or wrong

A - Incorrect: The trap is nonaccepting, so it does not accept.

B - Incorrect: A sink stays in place rather than restarting.

C - Correct: It remains in the trap for the remaining symbols and rejects when input ends.

D - Incorrect: The next transition returns to the trap itself.

Primary note proof

Study-note section AUT06

> A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0 \in Q$, accepting set $F \subseteq Q$ and total $\delta: Q \times \Sigma \to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition. F may be empty, recognizing $\varnothing$.

Study-note section AUT06

> Start at q_0 and consume the entire input. Accept iff the ending state is in F; visiting an accepting state earlier is insufficient. ε is accepted exactly when $q_0 \in F$. A nonaccepting sink loops on every symbol and cannot reach acceptance; it still finishes after finite input. Finite automata have finite memory encoded by states, not no state or no ability to remember any history.

Question sources

Automata & Complexity Theory review questions · Question 9

Question 110 | NS0175

The transition function of a DFA is defined as:

A. $\delta: Q \times \Sigma \to Q$

B. $\delta: Q \times \Sigma \to 2^Q$

C. $\delta: Q \times \Sigma \times \Gamma \to Q \times \Gamma^* \times \{L,R\}$

D. $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer A: $\delta:Q\times\Sigma\to Q$.

AUT06 defines a total function from a state/input-symbol pair to one state. A set-valued codomain describes NFA choices; allowing epsilon in the input domain describes epsilon-NFA transitions. Stack alphabets and write/direction outputs belong to richer models and are not components of the ordinary DFA transition signature.

Why each choice is right or wrong

A - Correct: This is the exact state/symbol to unique next-state signature of a DFA.

B - Incorrect: A powerset codomain allows nondeterministic next-state choices.

C - Incorrect: The extra storage alphabet and write/direction outputs do not describe an ordinary finite automaton.

D - Incorrect: Epsilon input and set-valued successors describe an epsilon-NFA rather than a DFA.

Primary note proof

Study-note section AUT06

> A deterministic finite automaton (DFA) is $M=(Q,\Sigma,\delta,q_0,F)$, with finite nonempty Q, start $q_0\in Q$, accepting set $F\subseteq Q$ and total $\delta:Q\times\Sigma\to Q$. Exactly one next state is specified per state/symbol pair; this does not mean the whole machine has one transition. F may be empty, recognizing $\varnothing$.

Study-note section AUT06

> Start at q_0 and consume the entire input. Accept iff the ending state is in F; visiting an accepting state earlier is insufficient. ε is accepted exactly when $q_0\in F$. A nonaccepting sink loops on every symbol and cannot reach acceptance; it still finishes after finite input. Finite automata have finite memory encoded by states, not no state or no ability to remember any history.

Question sources

Automata & Complexity Theory review questions · Question 10

Question 111 | NS0176

Which operation on regular languages gives all strings that are NOT in the original language?

A. Reverse (L^R)

B. Complement (Σ* - L)

C. Kleene Star (L*)

D. Concatenation

Hint

Track mandatory symbols and factors that may repeat or disappear. Use short test words to check both directions of each proposed language description.

Answer B: Complement, relative to the fixed alphabet universe.

AUT03 defines complement as $\Sigma^* - L$ and requires a fixed alphabet. This selects precisely the words in the universe absent from L. Reversal changes symbol order, star concatenates copies and concatenation joins words; none performs the required membership negation. Completing a DFA before swapping final states is the recognition construction in AUT14.

Why each choice is right or wrong

A - Incorrect: Reversal changes word order rather than selecting all words outside the language.

B - Correct: Complement contains exactly the universe's words that are not members of L.

C - Incorrect: Kleene star concatenates zero or more language words and does not negate membership.

D - Incorrect: Concatenation joins words from languages rather than forming the universe-relative exclusion.

Primary note proof

Study-note section AUT03

> Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, $L^*=\bigcup n\geq 0L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Study-note section AUT14

> Right-linear rules have form A→aB, A→a or permitted ε rules; left-linear rules put the single variable at the opposite end, A→Ba. Both consistent orientations generate REG; arbitrarily mixing them can generate nonregular languages, for example S→aA|ε, A→Sb generates a^nb^n. A CFG rule A→aBA is legal but not regular-grammar form.

Study-note section AUT14

> Regular languages are closed under union, concatenation, star, reversal, intersection, difference and complement.

Question sources

Automata & Complexity Theory review questions · Question 11

Question 112 | NS0177

The language L* (Kleene Star of L) is defined as:

A. $L^1 \cup L^2 \cup L^3 \cup ...$

B. $L \cap \Sigma^*$

C. All strings not in L

D. $L^0 \cup L^1 \cup L^2 \cup ...$ (includes empty string because $L^0 = \{\varepsilon\}$)

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer D: The union of all nonnegative language powers.

AUT03 defines L* by union over every n≥0, and AUT02 gives $L^0=\{\varepsilon\}$. Thus the zero power guarantees the empty word even when L is empty. A omits the zero power and instead describes positive closure; intersection with the universe and complement are different operations.

Why each choice is right or wrong

A - Incorrect: This omits L^0 and defines positive closure rather than general Kleene star.

B - Incorrect: Intersecting a language with its word universe simply retains L.

C - Incorrect: All words outside L define complement, not repeated concatenation.

D - Correct: The nonnegative-power union is the definition and its zeroth term contributes epsilon.

Primary note proof

Study-note section AUT03

> Σ^n contains words of exactly n symbols; for $|\Sigma|=k$, $|\Sigma^n|=k^n$, including $\Sigma^0=\{\varepsilon\}$. $\Sigma^*=\bigcup n\geq 0\Sigma^n$ and $\Sigma^+=\bigcup n\geq 1\Sigma^n$. Over a nonempty alphabet, Σ^+ excludes ε and $\Sigma^*=\Sigma^+\cup\{\varepsilon\}$. Σ^2 over {a,b} is {aa,ab,ba,bb}.

Study-note section AUT03

> For an arbitrary language L, $L^*=\bigcup n\geq 0 L^n$ and $L^+=LL^*$. Unlike alphabet positive closure, L^+ can contain ε when $\varepsilon\in L$. $\varnothing^*=\{\varepsilon\}$, $\varnothing^+=\varnothing$, and $\{\varepsilon\}^*=\{\varepsilon\}$. Union means membership in either set; intersection requires the same word in both, not concatenation. Complement is Σ^*-L and needs a fixed alphabet.

Study-note section AUT02

Concatenation xy writes x followed by y: cddc·ababb=cddcababb; order generally matters. εx=xε=x. Define $x^0=\varepsilon$ and x^n as n copies for n≥1. Reverse x^R reads symbols backward: caddba reverses to abddac; $(xy)^R=y^Rx^R$ and $(x^R)^R=x$.

Study-note section AUT02

For languages, $LM=\{xy:x\in L,y\in M\}$, $L^0=\{\varepsilon\}$, and $L^{n+1}=L^nL$. Thus {e,fgh}{c,d}={ec,ed,fghc,fghd}; fghe is absent.

Question sources

Automata & Complexity Theory review questions · Question 12

Question 113 | NS0178

In an NFA, the transition function maps each (state, input) pair to:

A. Exactly one next state

B. The initial state always

C. A SET of possible next states (including the empty set)

D. A symbol and a direction

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer C: A set of possible successors, possibly empty.

AUT09 defines an NFA transition as a mapping into the powerset of Q, so zero, one or several successors are permitted. The empty set represents a stuck branch, not an invalid machine. A uniquely determined singleton is only a deterministic special case; fixed resets and tape write/direction outputs do not describe the NFA signature.

Why each choice is right or wrong

A - Incorrect: One successor is permitted but not required; NFA transitions may have zero or several.

B - Incorrect: An NFA transition need not return to the initial state.

C - Correct: The powerset codomain permits any subset of states, including the empty set.

D - Incorrect: Write-symbol and direction outputs are tape-machine operations, not NFA state-set transitions.

Primary note proof

Study-note section AUT09

> A nondeterministic finite automaton (NFA) permits zero, one or many successors: $\delta:Q\times\Sigma\to P(Q)$. An ε-NFA additionally permits moves without consuming input, using domain $Q\times(\Sigma\cup\{\varepsilon\})$. Some books include ε moves in the term NFA; state the convention. An NFA accepts iff at least one finite path consumes the complete word and finishes in F, including permitted ε moves before/after symbols.

Study-note section AUT09

> An NFA is not more expressive than a DFA: both recognize exactly REG. Nondeterminism can make a description smaller. A DFA is a special NFA after interpreting its unique successor as a singleton set.

Question sources

Automata & Complexity Theory review questions · Question 13

Question 114 | NS0179

An NFA accepts a string if:

A. ALL computation paths lead to an accepting state

B. AT LEAST ONE computation path leads to an accepting state

C. The majority of paths lead to accepting states

D. The last state visited is the accepting state

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer B: At least one path consumes the whole word and finishes accepting.

AUT09 states existential acceptance over finite computation paths, with the entire word consumed and permitted epsilon moves included. B expresses that existential condition, interpreted with the full-input requirement. Failed branches do not cancel an accepting branch, and counting a majority or asking every path to accept gives different, incorrect conditions.

Why each choice is right or wrong

A - Incorrect: Requiring all paths would wrongly reject words with one accepting branch and another failing branch.

B - Correct: One complete accepting path suffices; it must consume the whole input.

C - Incorrect: Path majority is irrelevant; the acceptance condition is existential rather than statistical.

D - Incorrect: There is no single global last state across nondeterministic branches, and unread input cannot be ignored.

Primary note proof

Study-note section AUT09

> A nondeterministic finite automaton (NFA) permits zero, one or many successors: $\delta:Q\times\Sigma\to P(Q)$. An ε-NFA additionally permits moves without consuming input, using domain $Q\times(\Sigma\cup\{\varepsilon\})$. Some books include ε moves in the term NFA; state the convention. An NFA accepts iff at least one finite path consumes the complete word and finishes in F, including permitted ε moves before/after symbols.

Study-note section AUT09

> An NFA is not more expressive than a DFA: both recognize exactly REG. Nondeterminism can make a description smaller. A DFA is a special NFA after interpreting its unique successor as a singleton set.

Question sources

Automata & Complexity Theory review questions · Question 14

Question 115 | NS0180

Which statement about DFAs and NFAs is CORRECT?

A. Every NFA can be converted to an equivalent DFA recognizing the same language

B. NFAs are strictly more powerful than DFAs

C. DFAs can recognize languages that NFAs cannot

D. NFAs cannot recognize regular languages

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer A: Every NFA has an equivalent DFA.

AUT09 says both machine types recognize exactly regular languages, and AUT10 constructs equivalent DFA states as reachable NFA-state subsets. Nondeterminism may reduce representation size but does not add expressive power. Therefore A is true while each claim that one family recognizes languages inaccessible to the other is false.

Why each choice is right or wrong

A - Correct: Subset construction produces a DFA accepting exactly the NFA's language.

B - Incorrect: Nondeterminism may improve succinctness but does not exceed regular-language power.

C - Incorrect: Every DFA is also an NFA special case, so no such exclusive DFA language exists.

D - Incorrect: NFAs recognize exactly the regular languages, the opposite of this claim.

Primary note proof

Study-note section AUT09

> A nondeterministic finite automaton (NFA) permits zero, one or many successors: $\delta:Q\times\Sigma\to P(Q)$. An ε-NFA additionally permits moves without consuming input, using domain $Q\times(\Sigma\cup\{\varepsilon\})$. Some books include ε moves in the term NFA; state the convention. An NFA accepts iff at least one finite path consumes the complete word and finishes in F, including permitted ε moves before/after symbols.

Study-note section AUT09

> An NFA is not more expressive than a DFA: both recognize exactly REG. Nondeterminism can make a description smaller. A DFA is a special NFA after interpreting its unique successor as a singleton set.

Study-note section AUT10

> ε-closure(T) contains T and every state reachable through ε moves. Start the DFA at ε-closure($\{q_0\}$); for each current subset T and symbol a compute ε-closure($\bigcup q\in T\delta(q,a)$). A subset accepts iff it intersects F. Explore only reachable subsets; the empty subset is the rejecting sink when reached. An n-state NFA yields at most 2^n DFA states, not necessarily exactly 2^n.

Question sources

Automata & Complexity Theory review questions · Question 15

Question 116 | NS0181

What is an ε-transition in an NFA?

A. A transition that rejects the current input

B. A transition that only occurs at the end of input

C. A transition to the dead state

D. A transition that occurs without consuming any input symbol

Hint

Read the transition domain and acceptance condition carefully. Track which inputs are consumed and evaluate the whole word under the stated model.

Answer D: A move that consumes no input symbol.

AUT09 explicitly defines epsilon moves as transitions without consuming input. They may occur before, between or after symbol-consuming transitions when the machine permits them. They do not inherently mean rejection or movement to a sink, and they do not waive the requirement that an accepting computation consume the full word.

Why each choice is right or wrong

A - Incorrect: An epsilon move is not intrinsically rejecting; its destination and later path determine acceptance.

B - Incorrect: Permitted epsilon moves can occur before or between input symbols as well as at the end.

C - Incorrect: Its destination need not be a dead state.

D - Correct: The machine changes state without consuming a symbol from the input.

Primary note proof

Study-note section AUT09

> A nondeterministic finite automaton (NFA) permits zero, one or many successors: $\delta:Q\times\Sigma\to P(Q)$. An ε-NFA additionally permits moves without consuming input, using domain $Q\times(\Sigma\cup\{\varepsilon\})$. Some books include ε moves in the term NFA; state the convention. An NFA accepts iff at least one finite path consumes the complete word and finishes in F, including permitted ε moves before/after symbols.

Study-note section AUT09

> An NFA is not more expressive than a DFA: both recognize exactly REG. Nondeterminism can make a description smaller. A DFA is a special NFA after interpreting its unique successor as a singleton set.

Question sources

Automata & Complexity Theory review questions · Question 16

Question 117 | NS0182

When converting an NFA with n states to an equivalent DFA, the DFA may have at most how many states?

A. n states

B. n^2 states

C. 2^n states (the power set of Q)

D. n+1 states

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer C: At most 2^n states.

AUT10 states the subset-construction bound explicitly. An n-element state set has 2^n subsets, and each DFA state denotes one subset. Only reachable subsets need be included, so the bound need not be attained. Linear, quadratic and n+1 bounds are not general worst-case guarantees for NFA determinization.

Why each choice is right or wrong

A - Incorrect: A deterministic equivalent may require more than the original n states.

B - Incorrect: A quadratic bound does not cover the general powerset worst case.

C - Correct: There are 2^n possible subsets of the n NFA states.

D - Incorrect: An n+1 bound is not sufficient for general subset construction.

Primary note proof

Study-note section AUT10

> ε-closure(T) contains T and every state reachable through ε moves. Start the DFA at ε-closure({q_0}); for each current subset T and symbol a compute ε-closure(∪q∈Tδ(q,a)). A subset accepts iff it intersects F. Explore only reachable subsets; the empty subset is the rejecting sink when reached. An n-state NFA yields at most 2^n DFA states, not necessarily exactly 2^n.

Question sources

Automata & Complexity Theory review questions · Question 17

Question 118 | NS0183

What does the regular expression (0+1)*1 describe?

A. All binary strings

B. All binary strings that END with '1'

C. All binary strings starting with '1'

D. All binary strings containing exactly one '1'

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer B: All binary strings ending in 1.

With + as union, (0+1)* generates any finite binary prefix, including empty. Concatenating the final 1 forces the whole word to end in 1. It does not constrain the beginning or total number of ones: 01 and 11 are included, while epsilon and words ending in 0 are not.

Why each choice is right or wrong

A - Incorrect: Epsilon and words ending in zero are binary strings excluded by the final mandatory 1.

B - Correct: An arbitrary binary prefix followed by 1 is precisely a binary word ending in 1.

C - Incorrect: The arbitrary prefix may begin with zero, as in 01.

D - Incorrect: The prefix may contain additional ones, so exactly-one is too restrictive.

Primary note proof

Study-note section AUT12

> Classical expressions use ∅, ε, symbols, union, concatenation and star. A union B is written A|B or A+B; postfix r^+ means one-or-more r repetitions and is a different use of +. Parenthesize when conventions might clash. Star binds tightest, concatenation next, union last. r? means r|ε; r^+=rr*.

Study-note section AUT12

> (a|b)*a denotes words ending a; 0*10* denotes exactly one 1; a*(b|c) gives zero-or-more a's then exactly one b or c, so aab belongs but aaa and aaabc do not.

Question sources

Automata & Complexity Theory review questions · Question 18

Question 119 | NS0184

Which regular expression describes the language of all strings over {a,b} with at least one 'a'?

A. (a+b)*a(a+b)*

B. b*ab*

C. (a+b)+

D. a(a+b)*

Hint

Track mandatory symbols and factors that may repeat or disappear. Use short test words to check both directions of each proposed language description.

Answer A: (a+b)*a(a+b)*.

AUT13 gives this construction explicitly for at least one a. The middle a guarantees existence, while both stars permit additional a or b anywhere. b*ab* permits exactly one a; positive closure also permits all-b words, and a(a+b)* requires the first symbol rather than merely some symbol to be a.

Why each choice is right or wrong

A - Correct: The required middle a guarantees existence while unrestricted sides allow all other positions.

B - Incorrect: b*ab* excludes words with multiple a symbols and therefore denotes exactly one a.

C - Incorrect: Positive closure admits nonempty all-b words, which contain no a.

D - Incorrect: This requires a at the beginning and excludes valid words such as ba.

Primary note proof

Study-note section AUT13

> To require at least one a anywhere over {a,b}, use (a|b)*a(a|b)*. b*ab* requires exactly one a. At least two zeros uses (0|1)*0(0|1)*0(0|1)*; the zeros need not be adjacent. Testing 010 distinguishes it from a requirement for substring 00. For S→aS|ε, any number of a's is generated, giving a*.

Question sources

Automata & Complexity Theory review questions · Question 19

Question 120 | NS0185

The regular expression 0*10* represents:

A. All binary strings

B. All binary strings starting with 1

C. All binary strings containing exactly one '1'

D. All binary strings ending with '0'

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer C: Exactly one 1, with any number of zeros on either side.

AUT12 explicitly interprets 0*10* as exactly one 1. Both stars contribute only zeros, including none, so they cannot add another 1. The word 1 is accepted, while 11 and epsilon are not. No constraint forces the first symbol to be 1 or the final symbol to be zero.

Why each choice is right or wrong

A - Incorrect: Many binary words, including epsilon and 11, are excluded.

B - Incorrect: Leading zeros are permitted, so accepted words need not start with 1.

C - Correct: Only the mandatory middle symbol is 1; all other generated symbols are zeros.

D - Incorrect: The trailing zero star can be empty, so accepted words may end in 1.

Primary note proof

Study-note section AUT12

> Classical expressions use ∅, ε, symbols, union, concatenation and star. A union B is written A|B or A+B; postfix r^+ means one-or-more r repetitions and is a different use of +. Parenthesize when conventions might clash. Star binds tightest, concatenation next, union last. r? means r|ε; r^+=rr*.

Study-note section AUT12

> (a|b)*a denotes words ending a; 0*10* denotes exactly one 1; a*(b|c) gives zero-or-more a's then exactly one b or c, so aab belongs but aaa and aaabc do not.

Question sources

Automata & Complexity Theory review questions · Question 20

Question 121 | NS0186

Which of the following languages CANNOT be described by a regular expression?

A. All strings over {a,b} starting with 'a'

B. All binary strings of even length

C. All strings containing at least two 'b's

D. $\{a^n b^n \mid n \geq 1\}$ - strings with equal a's and b's

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer D: The equal-count language $a^n b^n$ is nonregular.

AUT15 uses $a^p b^p$ and shows that pumping any permitted initial a block breaks equal counts. Classical regular expressions cannot describe this language. Starting with a fixed symbol, even length and a minimum number of b occurrences require only finite-state information, so the other descriptions are regular.

Why each choice is right or wrong

A - Incorrect: A fixed starting symbol can be recognized by finitely many prefix states.

B - Incorrect: Even length needs only two parity states and is regular.

C - Incorrect: Counting b occurrences up to a threshold of two needs finitely many states.

D - Correct: Unbounded equality of separated counts defeats regular pumping and therefore classical regular expressions.

Primary note proof

Study-note section AUT15

> If L is regular, some $p \geq 1$ satisfies: every $s \in L$ with $|s| \geq p$ has a split $s=xyz$, $|xy| \leq p$, $|y| \geq 1$, and $xy^i z \in L$ for every $i \geq 0$.

Study-note section AUT15

> For $L=\{a^n b^n : n \geq 0\}$, choose $a^p b^p$. Every allowed $y=a^k$ with $k \geq 1$; pumping $i=0$ leaves fewer a's than b's. The contradiction excludes every DFA.

Question sources

Automata & Complexity Theory review questions · Question 21

Question 122 | NS0187

In the Chomsky hierarchy, which grammar type corresponds to Finite Automata?

A. Type 0 - Unrestricted Grammar

B. Type 3 - Regular Grammar

C. Type 2 - Context-Free Grammar

D. Type 1 - Context-Sensitive Grammar

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: Type 3 regular grammars.

AUT05 directly pairs Type 3 regular grammars with finite automata. Other grammar types can generate regular sublanguages, but the question asks which family corresponds to the finite-automaton model, an exact-family comparison. Type 2, Type 1 and Type 0 correspond to progressively stronger stack, bounded-tape and unrestricted-tape recognition families.

Why each choice is right or wrong

A - Incorrect: Type 0 corresponds to unrestricted Turing recognition rather than exactly finite-automaton power.

B - Correct: Type 3 is the regular grammar family exactly matched by finite automata.

C - Incorrect: Type 2 corresponds to nondeterministic pushdown recognition.

D - Incorrect: Type 1 corresponds to nondeterministic linear-bounded recognition.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Question sources

Automata & Complexity Theory review questions · Question 22

Question 123 | NS0188

A formal grammar is defined as a 4-tuple G = (V, Σ, P, S). What does 'P' represent?

A. The set of all accepting states

B. The power set of the terminal alphabet

C. The set of production rules (A → β)

D. The pushdown stack alphabet

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer C: The set of production rules.

AUT04 defines V as variables, Σ as terminals, P as productions and S as the start variable. Productions specify the allowed rewriting steps used to generate terminal words. Accepting states and stack alphabets belong to machine definitions, while an alphabet powerset is not the meaning of P in this grammar tuple.

Why each choice is right or wrong

A - Incorrect: Accepting states are a recognizer component, not the production component of a grammar.

B - Incorrect: P is not the powerset notation for the alphabet in this tuple.

C - Correct: P denotes the allowed grammar-production rewrite rules.

D - Incorrect: A stack alphabet belongs to PDA storage rather than the grammar's production set.

Primary note proof

Study-note section AUT04

> Write G=(V,Σ,P,S): finite variables V, terminals Σ disjoint from V, productions P and start variable S∈V. A production replaces its left side by its right side. The language is L(G)={w∈Σ*:S⇒*w}. A recognizer processes a word and accepts exactly the language members. Grammars generate languages; automata recognize languages, not grammars.

Study-note section AUT04

> Two grammars are equivalent when their generated languages are equal, regardless of rule counts or variable names. One common word, equal alphabets or one-way inclusion is insufficient.

Question sources

Automata & Complexity Theory review questions · Question 23

Question 124 | NS0189

In a RIGHT-LINEAR grammar, production rules have the form:

A. A → aB or A → a (terminal before optional non-terminal on right)

B. A → Ba or A → a (non-terminal before terminal)

C. A → α where α is any string of terminals and non-terminals

D. αAβ → αγβ

Hint

Track mandatory symbols and factors that may repeat or disappear. Use short test words to check both directions of each proposed language description.

Answer A: A→aB or A→a, with permitted ε rules where needed.

AUT14 places the optional single variable at the right end after the terminal. That gives A; B is the opposite left-linear orientation. An unrestricted string on the right is CFG form rather than necessarily regular, and a context-sensitive rewrite with surrounding context is a different grammar restriction. Epsilon conventions are handled separately.

Why each choice is right or wrong

A - Correct: The optional nonterminal follows the terminal at the right end of the production.

B - Incorrect: Putting the nonterminal before the terminal gives a left-linear orientation.

C - Incorrect: An arbitrary right-hand string can contain multiple variables and need not be regular-grammar form.

D - Incorrect: The context-sensitive contextual rewrite is not the defining right-linear production form.

Primary note proof

Study-note section AUT14

> Right-linear rules have form A→aB, A→a or permitted ε rules; left-linear rules put the single variable at the opposite end, A→Ba. Both consistent orientations generate REG; arbitrarily mixing them can generate nonregular languages, for example S→aA|ε, A→Sb generates $a^n b^n$. A CFG rule A→aBA is legal but not regular-grammar form.

Study-note section AUT14

> Regular languages are closed under union, concatenation, star, reversal, intersection, difference and complement.

Question sources

Automata & Complexity Theory review questions · Question 24

Question 125 | NS0190

Which production rule form is VALID in a Context-Free Grammar but NOT in a Regular Grammar?

A. A → a

B. A → aB

C. A → ε

D. A → aBA (two non-terminals on right-hand side)

Hint

Track mandatory symbols and factors that may repeat or disappear. Use short test words to check both directions of each proposed language description.

Answer D: A→aBA is context-free but not regular-grammar form.

AUT14 explicitly identifies A→aBA as legal CFG form but not regular form. A CFG restricts the left side to one variable while permitting a variable/terminal string on the right. The other choices are regular productions or permitted epsilon cases. Two right-hand nonterminals violate the regular linear restriction.

Why each choice is right or wrong

A - Incorrect: A→a is an ordinary regular production as well as context-free.

B - Incorrect: A→aB is exactly a right-linear regular production.

C - Incorrect: An epsilon rule is allowed under the note's regular-grammar convention.

D - Correct: The single-variable left side is CFG-valid, but two right-side variables violate regular linear form.

Primary note proof

Study-note section AUT14

> Right-linear rules have form A→aB, A→a or permitted ε rules; left-linear rules put the single variable at the opposite end, A→Ba. Both consistent orientations generate REG; arbitrarily mixing them can generate nonregular languages, for example S→aA|ε, A→Sb generates $a^n b^n$. A CFG rule A→aBA is legal but not regular-grammar form.

Study-note section AUT14

> Regular languages are closed under union, concatenation, star, reversal, intersection, difference and complement.

Question sources

Automata & Complexity Theory review questions · Question 25

Question 126 | NS0191

A grammar is said to be AMBIGUOUS if:

A. It generates infinitely many strings

B. Some string in the language has more than one parse tree

C. It contains left-recursive production rules

D. It uses both left-linear and right-linear rules

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: Some generated word has more than one parse tree.

AUT18 defines ambiguity by two distinct parse trees for at least one word, equivalently two leftmost derivations. Infinite size, left recursion and mixed linear orientations concern other properties and do not establish this condition. One must exhibit competing trees, not merely different expansion orders of a single tree.

Why each choice is right or wrong

A - Incorrect: Infinite languages can have unambiguous grammars, so infinitude does not define ambiguity.

B - Correct: Multiple parse trees for one generated word are the precise ambiguity criterion.

C - Incorrect: Left recursion can encode unambiguous left associativity; it does not alone prove ambiguity.

D - Incorrect: Mixing linear orientations concerns grammar form and is not the parse-tree multiplicity definition.

Primary note proof

Study-note section AUT18

> A CFG is ambiguous when some word has two distinct parse trees, equivalently two different leftmost derivations. Different expansion orders that give the same tree do not prove ambiguity.

Study-note section AUT18

> Separate precedence levels: E→E+T|T, T→T*F|F, F→id|(E). Multiplication binds inside T before addition at E, and left recursion encodes left association. Uniqueness follows by the top-level last operator split and induction. S→SS|ε is already ambiguous for ε. S→A|B is ambiguous if A→ab and B→abB|ε both produce ab. Ambiguous grammar and inherently ambiguous language differ: another grammar may remove ambiguity.

Question sources

Automata & Complexity Theory review questions · Question 26

Question 127 | NS0192

In a derivation tree (parse tree), what do the LEAF NODES represent?

A. Non-terminal symbols that haven't been expanded yet

B. The start symbol S

C. Terminal symbols - the actual characters in the generated string

D. ε-transitions in the grammar

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer C: Terminal symbols, with an explicit ε leaf where that convention is used.

AUT16 describes complete parse trees with terminal leaves and sometimes an epsilon leaf for an empty production. Reading leaves left to right yields the generated word. Unexpanded variables belong to incomplete trees, and a start symbol is normally the root. Epsilon productions are not automaton transitions in this representation.

Why each choice is right or wrong

A - Incorrect: Unexpanded nonterminals indicate an incomplete tree rather than leaves of the completed derivation.

B - Incorrect: S is normally the root, not the general category of terminal leaves.

C - Correct: Terminal leaves spell the derived word; epsilon may be displayed for an empty production.

D - Incorrect: A parse tree records productions, not automaton epsilon-transition edges.

Primary note proof

Study-note section AUT16

> In a context-free grammar (CFG) each production has one variable on the left, $A \to \alpha$ with $\alpha \in (V \cup \Sigma)^*$. The left side therefore also belongs to $V \cup \Sigma$ and its string closure; alternatives stating these weaker memberships can all be true. A sentential form is any string derived from S, including terminal-only sentences and S itself.

Study-note section AUT16

> A leftmost derivation always replaces the leftmost variable; a rightmost one replaces the rightmost variable. This is not scanning the completed input in opposite directions. A complete parse tree has root S, internal variable nodes expanded by productions and terminal leaves, with ε leaves when that convention is used. Read the leaves left-to-right for the yielded word. The whole tree represents derivation structure; a path in a derivation graph is a different object.

Question sources

Automata & Complexity Theory review questions · Question 27

Question 128 | NS0193

What is the key difference between left derivation and right derivation?

A. Left derivation always expands the LEFTMOST non-terminal; right derivation expands the RIGHTMOST

B. Left derivation produces left-linear grammars; right produces right-linear grammars

C. Left derivation is used only for regular grammars

D. They produce different final strings for the same grammar

Hint

Apply the stated transition, closure or equivalence rule step by step; distinguish a guaranteed bound from a value for one particular machine.

Answer A: Expand the leftmost versus rightmost remaining nonterminal.

AUT16 defines the two derivations by which remaining variable is rewritten. Both can describe the same final word and even the same parse tree. They do not determine whether the grammar itself is left-linear or right-linear, and the distinction applies to CFGs generally rather than only regular grammars.

Why each choice is right or wrong

A - Correct: The choice of the leftmost or rightmost remaining variable is the defining difference.

B - Incorrect: Derivation order does not convert the production system into a left-linear or right-linear grammar.

C - Incorrect: Leftmost derivation is used for context-free grammars generally, not only regular ones.

D - Incorrect: Different expansion orders can generate the same terminal word and the same tree.

Primary note proof

Study-note section AUT16

> In a context-free grammar (CFG) each production has one variable on the left, $A \to \alpha$ with $\alpha \in (V \cup \Sigma)^*$. The left side therefore also belongs to $V \cup \Sigma$ and its string closure; alternatives stating these weaker memberships can all be true. A sentential form is any string derived from S, including terminal-only sentences and S itself.

Study-note section AUT16

> A leftmost derivation always replaces the leftmost variable; a rightmost one replaces the rightmost variable. This is not scanning the completed input in opposite directions. A complete parse tree has root S, internal variable nodes expanded by productions and terminal leaves, with ε leaves when that convention is used. Read the leaves left-to-right for the yielded word. The whole tree represents derivation structure; a path in a derivation graph is a different object.

Question sources

Automata & Complexity Theory review questions · Question 28

Question 129 | NS0194

The language $\{a^n b^n \mid n \geq 1\}$ is:

A. Regular - recognized by a DFA

B. Not a formal language

C. Context-Free but NOT Regular - recognized by a PDA, not a DFA

D. Context-Sensitive but not Context-Free

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer C: Context-free but not regular.

AUT17 generates $a^n b^n$ with matched recursive additions, using ab as the base for n≥1. AUT15 proves the family nonregular by pumping the initial a block. Thus a PDA suffices but a DFA does not. It remains a formal language and is also CSL, so a description denying context-freeness is wrong.

Why each choice is right or wrong

A - Incorrect: The regular pumping argument rules out any DFA for unbounded equal counts.

B - Incorrect: This is a precisely specified set of words and therefore a formal language.

C - Correct: A CFG/PDA handles matched counts, while the language is not regular.

D - Incorrect: It is context-sensitive too, but denying its context-free status is false.

Primary note proof

Study-note section AUT17

> Use a recursive rule to enforce matched structure and a base rule to stop. $S \to aSb \mid \varepsilon$ generates exactly $a^n b^n$ for n≥0: n recursive steps add n a's on the left and n b's on the right. Derive aabb by $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$. For n≥1 replace ε by base ab.

Study-note section AUT15

> If L is regular, some p≥1 satisfies: every $s \in L$ with $|s| \geq p$ has a split $s=xyz$, $|xy| \leq p$, $|y| \geq 1$, and $xy^i z \in L$ for every i≥0.

Study-note section AUT15

> For $L=\{a^n b^n : n \geq 0\}$, choose $a^p b^p$. Every allowed $y=a^k$ with k≥1; pumping i=0 leaves fewer a's than b's. The contradiction excludes every DFA.

Study-note section AUT05

Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Question sources

Automata & Complexity Theory review questions · Question 29

Question 130 | NS0195

Every regular language is a context-free language. This means:

A. Every CFL can be described by a regular expression

B. Any language recognizable by a DFA/NFA can also be recognized by a PDA

C. Regular grammars are more powerful than CFGs

D. CFGs and regular grammars generate the same set of languages

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: A PDA can also recognize every DFA/NFA language.

AUT05 places regular languages inside context-free languages, and AUT09 equates DFA with NFA power. Thus a PDA can recognize every finite-automaton language, for example by ignoring its extra stack. The containment is strict; it does not give regular expressions for every CFL or make regular and CFG language families equal.

Why each choice is right or wrong

A - Incorrect: The proper inclusion leaves CFLs that no classical regular expression describes.

B - Correct: Finite-automaton languages are included in the PDA-recognizable context-free family.

C - Incorrect: Regular grammars are less expressive, not more expressive, than CFGs.

D - Incorrect: The note gives a proper containment rather than equality of the two families.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Question sources

Automata & Complexity Theory review questions · Question 30

Question 131 | NS0196

What additional component does a Pushdown Automaton (PDA) have that a Finite Automaton lacks?

A. A set of accepting states

B. A transition function

C. An input tape

D. A stack (LIFO memory)

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer D: A stack.

AUT21 defines a PDA by adding a stack to finite control, with its own alphabet and initial symbol. Finite automata already have states, accepting states and transition rules and process input. The distinguishing unbounded LIFO storage is what supports matched-count recognition beyond finite-state memory.

Why each choice is right or wrong

A - Incorrect: Finite automata also have an accepting-state set.

B - Incorrect: Finite automata already use a transition function.

C - Incorrect: Both models process input; an input representation is not the added PDA memory.

D - Correct: The stack is the extra LIFO storage that distinguishes the PDA from finite control alone.

Primary note proof

Study-note section AUT21

> A nondeterministic pushdown automaton (PDA) adds a stack to finite control: $(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$. Its transition relation maps (state,input symbol or ε,stack top) to possible (new state,replacement stack string) pairs.

Study-note section AUT-S02

> A stack is last-in, first-out: push adds an item at the top and pop removes the top item, so the most recently pushed item is removed first. A PDA accesses this top rather than an arbitrary interior position.

Study-note section AUT-S02

> A queue is first-in, first-out: insertion at the rear and removal at the front returns the oldest waiting item. A linked list can implement a storage discipline, but the implementation container is different from the abstract push/pop or enqueue/dequeue access rule.

Question sources

Automata & Complexity Theory review questions · Question 31

Question 132 | NS0197

The stack in a PDA operates as a:

A. Last-In-First-Out (LIFO) data structure

B. First-In-First-Out (FIFO) data structure

C. Random access memory

D. Circular queue

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Last in, first out.

AUT21 defines each PDA move using the current stack top and its replacement string. This top-only discipline means the most recently pushed item is the next removed: LIFO. A queue removes oldest items, random-access storage permits arbitrary addressing, and circular-queue organization is not the PDA stack model.

Why each choice is right or wrong

A - Correct: Top-only push/pop access removes the most recently pushed item first.

B - Incorrect: FIFO removes the oldest item and describes queue rather than stack behavior.

C - Incorrect: A PDA cannot arbitrarily access interior stack positions like random-access memory.

D - Incorrect: Circular queue organization does not express the top-only LIFO stack discipline.

Primary note proof

Study-note section AUT21

> A nondeterministic pushdown automaton (PDA) adds a stack to finite control: $(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$. Its transition relation maps (state,input symbol or ε,stack top) to possible (new state,replacement stack string) pairs.

Study-note section AUT-S02

> A stack is last-in, first-out: push adds an item at the top and pop removes the top item, so the most recently pushed item is removed first. A PDA accesses this top rather than an arbitrary interior position.

Study-note section AUT-S02

> A queue is first-in, first-out: insertion at the rear and removal at the front returns the oldest waiting item. A linked list can implement a storage discipline, but the implementation container is different from the abstract push/pop or enqueue/dequeue access rule.

Question sources

Automata & Complexity Theory review questions · Question 32

Question 133 | NS0198

A PDA formally defined as a 7-tuple includes which of the following that a DFA does NOT have?

A. A set of states Q

B. An input alphabet Σ

C. A stack alphabet Γ and initial stack symbol Z

D. A set of accepting states F

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: A stack alphabet and initial stack symbol.

AUT21 gives the PDA tuple $(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$. Compared with a DFA, Γ and Z_0 describe its added stack storage. States, input symbols and accepting states occur in both models. Their presence is not what expands the recognizer beyond finite-state control.

Why each choice is right or wrong

A - Incorrect: Both DFA and PDA models have a finite control-state set Q.

B - Incorrect: Both machines process symbols from an input alphabet Σ.

C - Correct: Γ and Z_0 specify stack storage, which an ordinary DFA lacks.

D - Incorrect: Accepting-state sets occur in both models and do not distinguish stack memory.

Primary note proof

Study-note section AUT21

> A nondeterministic pushdown automaton (PDA) adds a stack to finite control: $(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$. Its transition relation maps (state,input symbol or ε,stack top) to possible (new state,replacement stack string) pairs. Fix which end denotes the top before tracing. Replacing X by X leaves the stack unchanged; ε input means consume no symbol, while replacing X by ε pops it. These are different uses of ε.

Question sources

Automata & Complexity Theory review questions · Question 33

Question 134 | NS0199

The transition function of a PDA takes which inputs and produces which outputs?

A. (state, input) → next state

B. (state, input symbol or ε, top of stack) → (new state, new stack string)

C. (state, input) → set of states

D. (state, stack top, next input) → (write symbol, direction)

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: State, input-or-ε and stack top map to possible new states and replacement stack strings.

AUT21 defines a PDA move by the current state, consumed symbol or epsilon and top stack symbol. Its possible outputs combine a new state with a replacement stack string. Set-valued nondeterminism is understood even though B abbreviates an individual result. The alternatives lacking stack access or using tape directions describe other models.

Why each choice is right or wrong

A - Incorrect: This omits the stack top and replacement string essential to PDA behavior.

B - Correct: This includes the current stack top and replacement string; nondeterminism permits a set of such results.

C - Incorrect: A state-set output without stack operations describes finite-state nondeterminism instead.

D - Incorrect: Writing a tape symbol and moving a head is TM-style machinery, not PDA stack replacement.

Primary note proof

Study-note section AUT21

> A nondeterministic pushdown automaton (PDA) adds a stack to finite control: $(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$. Its transition relation maps (state,input symbol or ε,stack top) to possible (new state,replacement stack string) pairs. Fix which end denotes the top before tracing. Replacing X by X leaves the stack unchanged; ε input means consume no symbol, while replacing X by ε pops it. These are different uses of ε.

Question sources

Automata & Complexity Theory review questions · Question 34

Question 135 | NS0200

Which of the following languages can a PDA recognize but a DFA CANNOT?

A. All binary strings ending with '1'

B. All strings over {a,b} starting with 'ab'

C. $\{a^n b^n \mid n \geq 1\}$ - strings with equal number of a's and b's

D. All strings of even length over {a,b}

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer C: The nonregular equal-count language $a^n b^n$.

AUT22 supplies the push-one/pop-one PDA procedure for equal a/b counts, and AUT15 excludes a DFA. Fixed suffixes, fixed prefixes and length parity require only finite memory, so both machine types can recognize those. C alone describes a language in PDA capability but outside DFA capability.

Why each choice is right or wrong

A - Incorrect: A fixed final symbol is a regular condition recognized by a DFA.

B - Incorrect: A fixed starting prefix needs only finitely many match stages and is regular.

C - Correct: A PDA matches unbounded equal counts using its stack, while regular pumping rules out a DFA.

D - Incorrect: Even length needs only a finite parity state, so a DFA already suffices.

Primary note proof

Study-note section AUT22

> For $a^n b^n$, push one X per a, switch to a b-reading phase, pop one X per b, and accept when input ends at the bottom marker. Reject a after entering the b phase and b when no X remains. Handle ε explicitly if n=0 is permitted. A stack counts without storing the full input as random-access memory.

Study-note section AUT15

> If L is regular, some $p \geq 1$ satisfies: every $s \in L$ with $|s| \geq p$ has a split $s=xyz$, $|xy| \leq p$, $|y| \geq 1$, and $xy^i z \in L$ for every $i \geq 0$.

Study-note section AUT15

> For $L=\{a^n b^n : n \geq 0\}$, choose $a^p b^p$. Every allowed $y=a^k$ with $k \geq 1$; pumping i=0 leaves fewer a's than b's. The contradiction excludes every DFA.

Question sources

Automata & Complexity Theory review questions · Question 35

Question 136 | NS0201

The language $\{a^n b^n c^n \mid n \geq 1\}$ is NOT a context-free language. This means:

A. It can be recognized by a PDA

B. It can be described by a context-free grammar

C. It is not a formal language

D. No PDA can recognize it; it requires at least a Linear Bounded Automaton (LBA)

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer D: No PDA recognizes it; an LBA suffices in the listed hierarchy.

AUT24 proves three equal counts non-context-free, and AUT25 recognizes them by marking matched triples within the input region. Thus no ordinary single-stack PDA recognizes the language, while an LBA does. It is still a precisely defined formal language; non-context-freeness is a family boundary rather than failure to be a language.

Why each choice is right or wrong

A - Incorrect: PDA-recognizable languages are context-free, contradicting the stated non-CFL property.

B - Incorrect: A CFG would make it context-free, which the pumping proof rules out.

C - Incorrect: It remains a formal language despite lying outside the context-free family.

D - Correct: No PDA suffices; bounded triple marking provides an LBA recognizer in the next listed family.

Primary note proof

Study-note section AUT25

> A context-sensitive rule $\alpha A \beta \to \alpha \gamma \beta$ replaces A in context, with γ nonempty. Equivalently, noncontracting grammar presentations preserve/increase total length, with the isolated ε exception. The condition is on the complete sides, not an unrelated comparison $|\beta| \geq |\alpha|$.

Study-note section AUT25

> A linear bounded automaton (LBA) is a TM with O(n) tape cells on length-n input; the standard CSL characterization uses nondeterministic LBAs. For $a^n b^n c^n$ first verify a*b*c* order, repeatedly mark one unmarked a, b and c, and reject missing/excess symbols.

Study-note section AUT25

> Thus every CSL is decidable.

Study-note section AUT24

If L is context-free, some p ensures each sufficiently long $s \in L$ has $s=uvxyz$ with $|vxy| \le p$, $|vy| \ge 1$ and $uv^ixy^iz \in L$ for every $i \ge 0$. The two pieces v and y are pumped together; either may be empty, not both. A non-CFL proof must defeat every allowed decomposition.

Study-note section AUT24

For $s=a^pb^pc^p$, the window vxy cannot contain both an a and a c. Pumping affects at most two blocks and at least one count. Use i=0 or 2: the unaffected third block retains p while another count changes, or symbol order breaks. Hence $a^nb^nc^n$ is not CFL.

Question sources

Automata & Complexity Theory review questions · Question 36

Question 137 | NS0202

In a Context-Sensitive Grammar, the production rule $\alpha A \beta \rightarrow \alpha \gamma \beta$ requires:

A. $|\gamma| < |A|$ (right side shorter than left)

B. $|\alpha\gamma\beta| \geq |\alpha A \beta|$ (right side no shorter than left side)

C. α and β must be empty strings

D. γ must be a single terminal symbol

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: The complete right-hand side cannot be shorter.

AUT25 explains contextual rules with a nonempty replacement γ, equivalently noncontracting full-side length under the isolated epsilon convention. The comparison must include the complete left and right sides, making B correct. Context α and β need not vanish, and γ may contain more than a single terminal.

Why each choice is right or wrong

A - Incorrect: A shorter replacement contradicts the noncontracting context-sensitive requirement.

B - Correct: The full right-hand side must be at least as long as the full left-hand side.

C - Incorrect: Context can be present; requiring empty α and β is not the context-sensitive rule.

D - Incorrect: Gamma is nonempty but need not be exactly one terminal.

Primary note proof

Study-note section AUT25

> A context-sensitive rule $\alpha A \beta \rightarrow \alpha \gamma \beta$ replaces A in context, with γ nonempty. Equivalently, noncontracting grammar presentations preserve/increase total length, with the isolated ε exception. The condition is on the complete sides, not an unrelated comparison $|\beta| \geq |\alpha|$.

Study-note section AUT25

> A linear bounded automaton (LBA) is a TM with O(n) tape cells on length-n input; the standard CSL characterization uses nondeterministic LBAs. For $a^n b^n c^n$ first verify a*b*c* order, repeatedly mark one unmarked a, b and c, and reject missing/excess symbols.

Study-note section AUT25

> Thus every CSL is decidable.

Question sources

Automata & Complexity Theory review questions · Question 37

Question 138 | NS0203

The Turing Machine transition function δ(q, a) = (p, b, L) means:

A. In state q reading symbol a: go to state p, write b, move head LEFT

B. In state q reading symbol a: go to state p, write b, move head RIGHT

C. In state q reading symbol a: go to state p, push b onto stack, move LEFT

D. In state q reading symbol a: delete b, move LEFT, enter state p

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Write b, move left, and enter p after reading a in q.

AUT27 directly interprets δ(q,a)=(p,b,L). The current scanned symbol a is replaced with b, the head shifts one cell left and control enters p. L fixes the direction; there is no stack push operation, and deletion is not the meaning of writing b.

Why each choice is right or wrong

A - Correct: This exactly preserves the specified new state, written symbol and leftward head movement.

B - Incorrect: R would indicate right movement; the given L specifies left.

C - Incorrect: The TM writes b on tape rather than pushing onto a separate stack.

D - Incorrect: The transition writes b; it does not mean delete b.

Primary note proof

Study-note section AUT27

> δ(q,a)=(p,b,L) means read a in q, write b, move one cell left and enter p. Accept/reject halting states have no further moves under this convention.

Question sources

Automata & Complexity Theory review questions · Question 38

Question 139 | NS0204

Which of the following is TRUE about Turing Machines?

A. A TM has a finite tape of fixed length

B. A TM can only move the head in one direction

C. A TM can simulate any algorithm - it is the most powerful computational model

D. A TM is equivalent in power to a PDA

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer C: A TM models general effective algorithmic computation.

AUT28 identifies Turing completeness with simulating general TM computation and relates effective procedures to the Church-Turing thesis. C expresses this ordinary model comparison, not omnipotence over undecidable tasks or a speed guarantee. A standard tape is unbounded, heads may move both directions, and TMs exceed single-stack PDA recognition power.

Why each choice is right or wrong

A - Incorrect: A standard TM tape is unbounded rather than fixed to a finite length.

B - Incorrect: Standard TM transitions may move the head left or right.

C - Correct: Under the effective-algorithm interpretation, TM computation captures general algorithmic procedures.

D - Incorrect: TMs recognize languages beyond the context-free languages of ordinary PDAs.

Primary note proof

Study-note section AUT28

> An off-line TM has a read-only input tape and separate read/write work tape. Multitape, multihead, two-way-infinite-tape and nondeterministic TMs have the same language-recognition power as standard TMs, though simulation costs differ. Two unbounded stacks can simulate a tape. Turing completeness means ability to simulate general TM computation, not polynomial speed or literally using infinitely many cells in a finite run. The Church-Turing thesis connects effective procedures to this formal model; it is not a proved runtime theorem.

Question sources

Automata & Complexity Theory review questions · Question 39

Question 140 | NS0205

Which option summarizes the pumping lemma for regular languages, rather than a separate consequence for a chosen recognizing DFA?

A. Every string in L can be pumped to produce longer strings still in L

B. Any sufficiently long string s in L can be split as xyz where the middle part y can be 'pumped' (repeated any number of times) and stay in L

C. L is infinite if and only if a string in L is longer than the number of states

D. All strings in L have the same length

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer B: Any sufficiently long string s in L can be split as xyz where the middle part y can be 'pumped' (repeated any number of times) and stay in L

For each regular language there is a pumping length p. Every accepted word of length at least p has a decomposition xyz with $|xy| \le p$ and $|y| > 0$ such that xy^iz remains accepted for every $i \ge 0$. The threshold and nonempty repeated part matter; short words need not be pumpable. The infinitude criterion involving a DFA's state count is a separate consequence, while a regular language can contain words of many different lengths.

Why each choice is right or wrong

A - Incorrect: The lemma applies only above a length threshold and pumps a suitable part, not every word without qualification.

B - Correct: This states the sufficiently-long-word split and repeatable middle part; the formal bounds are supplied in the explanation.

C - Incorrect: A fixed-DFA infinitude criterion is a consequence rather than the pumping lemma's quantified statement.

D - Incorrect: Regular languages need not have constant word length; 0* contains arbitrarily many lengths.

Primary note proof

Study-note section AUT15

> If L is regular, some $p \ge 1$ satisfies: every $s \in L$ with $|s| \ge p$ has a split $s=xyz$, $|xy| \le p$, $|y| \ge 1$, and $xy^iz \in L$ for every $i \ge 0$.

Study-note section AUT15

> For $L=\{a^nb^n : n \ge 0\}$, choose a^pb^p. Every allowed $y=a^k$ with $k \ge 1$; pumping $i=0$ leaves fewer a's than b's. The contradiction excludes every DFA.

Question sources

Automata & Complexity Theory review questions · Question 40

Question 141 | NS0206

The language $L = \{a^n b^n c^n \mid n \geq 1\}$ is NOT context-free. Which technique proves this?

A. The Pumping Lemma for Regular Languages

B. The Chomsky hierarchy alone

C. Constructing a PDA and showing it fails

D. The Pumping Lemma for Context-Free Languages - any pumping violates the equal-count constraint

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer D: The CFL pumping lemma, defeating every permitted decomposition.

AUT24 chooses $a^p b^p c^p$ and shows the bounded vxy window cannot cover both a and c blocks. Pumping changes at most two counts or breaks block order, defeating every allowed decomposition. A failed individual PDA is not a proof against all PDAs, and regular pumping excludes only regularity, not the stronger CFL property.

Why each choice is right or wrong

A - Incorrect: Regular pumping disproves regularity and does not by itself establish non-context-freeness.

B - Incorrect: The hierarchy alone does not prove the specific language lies outside CFL.

C - Incorrect: Failure of one proposed PDA does not rule out every possible PDA.

D - Correct: The CFL pumping proof breaks every legal decomposition by changing counts or symbol order.

Primary note proof

Study-note section AUT24

> If L is context-free, some p ensures each sufficiently long $s \in L$ has $s=uvxyz$ with $|vxy| \leq p$, $|vy| \geq 1$ and $uv^i xy^i z \in L$ for every $i \geq 0$. The two pieces v and y are pumped together; either may be empty, not both. A non-CFL proof must defeat every allowed decomposition.

Study-note section AUT24

> For $s=a^p b^p c^p$, the window vxy cannot contain both an a and a c. Pumping affects at most two blocks and at least one count. Use i=0 or 2: the unaffected third block retains p while another count changes, or symbol order breaks. Hence $a^n b^n c^n$ is not CFL.

Question sources

Automata & Complexity Theory review questions · Question 41

Question 142 | NS0207

'Turing Completeness' means a system:

A. Can simulate a Turing Machine and thus compute anything computable

B. Solves problems in polynomial time

C. Uses an infinite amount of memory

D. Is equivalent to a Pushdown Automaton

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Ability to simulate general Turing-machine computation.

AUT28 defines Turing completeness by general TM simulation. This says which effective computations can be expressed, not whether they run in polynomial time. A finite execution uses finitely many cells even though the model permits unbounded memory. Single-stack PDA equivalence is weaker than Turing universality.

Why each choice is right or wrong

A - Correct: Simulation of general TM computation is the definition of Turing completeness.

B - Incorrect: Universality does not guarantee a polynomial-time algorithm for every computable problem.

C - Incorrect: A finite run need not and cannot literally use infinitely many cells.

D - Incorrect: An ordinary PDA has less expressive recognition power than a TM.

Primary note proof

Study-note section AUT28

> An off-line TM has a read-only input tape and separate read/write work tape. Multitape, multihead, two-way-infinite-tape and nondeterministic TMs have the same language-recognition power as standard TMs, though simulation costs differ. Two unbounded stacks can simulate a tape. Turing completeness means ability to simulate general TM computation, not polynomial speed or literally using infinitely many cells in a finite run. The Church-Turing thesis connects effective procedures to this formal model; it is not a proved runtime theorem.

Question sources

Automata & Complexity Theory review questions · Question 42

Question 143 | NS0208

The complexity class P contains problems that are:

A. Hard to solve and hard to verify

B. Solvable in polynomial time by a deterministic Turing Machine

C. Solvable only by nondeterministic machines

D. Requiring exponential space to solve

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer B: Deterministic polynomial-time decidable problems.

AUT34 defines P by deterministic polynomial-time decision relative to input length. It is a solvability upper bound, not a requirement of hardness, nondeterminism or exponential space. A problem in P may also have nondeterministic algorithms, but it is the existence of a deterministic polynomial-time decider that establishes membership.

Why each choice is right or wrong

A - Incorrect: P is defined by efficient deterministic decision, not by hardness of solving or verification.

B - Correct: This is the exact deterministic polynomial-time definition of P.

C - Incorrect: Problems in P have deterministic algorithms and are not solvable only nondeterministically.

D - Incorrect: Exponential space is not a requirement for a polynomial-time computation.

Primary note proof

Study-note section AUT34

> Complexity classes here contain decision languages. P consists of deterministic polynomial-time decidable languages; NP consists of nondeterministic polynomial-time decidable languages, equivalently polynomially verifiable yes-instances with polynomial-size certificates. A verifier must accept some certificate exactly for yes-inputs; no certificate may falsely establish a no-instance. NP means nondeterministic polynomial, not non-polynomial.

Study-note section AUT34

> P⊆NP because a verifier can ignore its certificate and run the P algorithm. P=NP remains open; NP does not mean definitely hard or definitely outside P.

Question sources

Automata & Complexity Theory review questions · Question 43

Question 144 | NS0209

The key property of problems in class NP is:

A. They can be solved in polynomial time by a deterministic TM

B. They require exponential time to verify solutions

C. Given a potential solution, it can be VERIFIED in polynomial time

D. They require polynomial space but exponential time

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer C: Polynomial-time verification of polynomial-size yes-certificates.

AUT34 equates NP with polynomially verifiable yes-instances using polynomial-size certificates. A verifier must not certify no-instances. Polynomial-time deterministic solving defines P and is not known for every NP problem; exponential verification or execution bounds are not part of NP's defining certificate condition.

Why each choice is right or wrong

A - Incorrect: Deterministic polynomial-time solution defines P; whether it covers all NP is unresolved.

B - Incorrect: NP verification is polynomial for an appropriate certificate rather than requiring exponential time.

C - Correct: A polynomial-size yes-certificate can be checked in polynomial time, the equivalent NP definition.

D - Incorrect: NP does not require exponential running time; it includes the polynomial-time languages.

Primary note proof

Study-note section AUT34

> Complexity classes here contain decision languages. P consists of deterministic polynomial-time decidable languages; NP consists of nondeterministic polynomial-time decidable languages, equivalently polynomially verifiable yes-instances with polynomial-size certificates. A verifier must accept some certificate exactly for yes-inputs; no certificate may falsely establish a no-instance. NP means nondeterministic polynomial, not non-polynomial.

Study-note section AUT34

> P⊆NP because a verifier can ignore its certificate and run the P algorithm. P=NP remains open; NP does not mean definitely hard or definitely outside P.

Question sources

Automata & Complexity Theory review questions · Question 44

Question 145 | NS0210

If a problem is NP-COMPLETE, it means:

A. It is in NP but not in P

B. It is the easiest problem in NP

C. It is NP-hard but might not be in NP

D. It is both in NP AND NP-hard; every problem in NP reduces to it in polynomial time

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer D: Both NP membership and NP-hardness under polynomial reductions.

AUT36 defines NP-complete as being in NP and hard for every NP language under polynomial reduction. It does not assert exclusion from P, since P versus NP is open. NP-hard alone need not even be decidable, while complete problems are the hardest in NP under the stated reduction, not its easiest members.

Why each choice is right or wrong

A - Incorrect: NP-completeness does not prove exclusion from P; that would settle the open P-versus-NP question.

B - Incorrect: Completeness concerns hardness for the whole class, not easiest membership.

C - Incorrect: This describes NP-hardness without the required NP-membership condition.

D - Correct: Both membership and polynomial-reduction hardness for every NP problem are required.

Primary note proof

Study-note section AUT36

> X is NP-hard if every $A \in NP$ satisfies $A \leq_p X$. It need not belong to NP and can even be undecidable. X is NP-complete if $X \in NP$ and X is NP-hard. NP-complete languages lie inside NP and can be decided by finite exhaustive certificate search in exponential time. They are not known to require exponential time.

Question sources

Automata & Complexity Theory review questions · Question 45

Question 146 | NS0211

Which is an established reason for the foundational significance of Boolean satisfiability (SAT) in NP-completeness theory?

A. It was the FIRST problem proven NP-complete (Cook-Levin theorem); every NP problem reduces to SAT

B. It can be solved in polynomial time

C. It is in class P

D. It requires exponential space to verify

Hint

Distinguish a foundational hardness proof, efficient certificate verification and an unproved efficient solving algorithm.

Answer A: It was the FIRST problem proven NP-complete (Cook-Levin theorem); every NP problem reduces to SAT

The Cook-Levin theorem made SAT the first problem proved NP-complete: its proof encodes an arbitrary nondeterministic polynomial-time computation as a Boolean formula. Hence every NP decision problem has a polynomial-time reduction to SAT. This starts the chain of later hardness reductions. A satisfying assignment can be checked in polynomial time. Whether SAT itself has a deterministic polynomial-time algorithm is unresolved, so neither membership in P nor such an algorithm is an established reason.

Why each choice is right or wrong

A - Correct: Cook-Levin established the first NP-complete problem and reductions from every problem in NP.

B - Incorrect: A general deterministic polynomial-time SAT algorithm is not established; one would imply P=NP.

C - Incorrect: SAT membership in P is unresolved, so this is not an established statement.

D - Incorrect: An assignment can be verified in polynomial time, without requiring exponential verification space.

Primary note proof

Study-note section AUT-S03

> Boolean satisfiability was the first problem proved NP-complete, through the Cook-Levin theorem. Its hardness proof starts directly from an arbitrary nondeterministic polynomial-time computation and constructs a Boolean formula that is satisfiable exactly when an accepting computation exists.

Study-note section AUT-S03

> This starting point matters: once one NP-complete problem is available, later hardness proofs can reduce that known problem to a new target. Membership in NP remains a separate

obligation. Checking a satisfying assignment is efficient; whether finding one is always polynomial-time remains unresolved.

Study-note section AUT36

SAT asks whether a Boolean formula has a satisfying assignment; Cook-Levin establishes SAT NP-complete. CNF-SAT and 3-SAT are NP-complete versions.

Question sources

Automata & Complexity Theory review questions · Question 46

Question 147 | NS0212

Which containment chain is established unconditionally, without assuming answers to the open equalities among these classes? Here EXP means EXPTIME.

A. NP ⊆ P ⊆ PSPACE ⊆ EXP

B. P ⊆ NP ⊆ PSPACE ⊆ EXP

C. PSPACE ⊆ NP ⊆ P ⊆ EXP

D. EXP ⊆ PSPACE ⊆ NP ⊆ P

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer B: P⊆NP⊆PSPACE⊆EXPTIME.

The established resource-class chain is P⊆NP⊆PSPACE⊆EXPTIME. These subset relations need not all be strict: the adjacent equalities named in the note remain unresolved. A reverse relation such as NP⊆P cannot therefore be selected as an unconditional theorem. Reversing the entire chain would additionally put EXPTIME inside P, contradicting the known strict separation P⊊EXPTIME. The revised scope asks for proved inclusion, not for a speculative equality.

Why each choice is right or wrong

A - Incorrect: This includes NP⊆P, which would settle the open P-versus-NP question rather than state a known unconditional chain.

B - Correct: P⊆NP⊆PSPACE⊆EXPTIME is the established chain.

C - Incorrect: This requires reverse containments that are not established unconditionally.

D - Incorrect: Reversing the entire chain would imply EXPTIME⊆P, contradicting the known strict time-hierarchy separation.

Primary note proof

Study-note section AUT39

> Use L⊆NL⊆P⊆NP⊆PSPACE⊆EXPTIME⊆EXPSPACE. Here L/NL use logarithmic workspace; EXPTIME is ⋃kTIME(2^(n^k)), an upper-bound class that also contains P, not a statement that every member requires exponential time.

Study-note section AUT39

> P versus NP, NP versus PSPACE and PSPACE versus EXPTIME remain unresolved. coNP contains complements of NP languages; P⊆NP∩coNP, and NP=coNP is open.

Question sources

Automata & Complexity Theory review questions · Question 47

Question 148 | NS0213

If problem A is REDUCIBLE to problem B in polynomial time ($A \leq_p B$), what can we conclude?

A. A is at least as hard as B

B. A and B are in the same complexity class

C. B is at least as hard as A; an efficient algorithm for B gives an efficient algorithm for A

D. B can be solved in the same time as A

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer C: B is at least as hard as A; solving B efficiently solves A efficiently.

AUT35 defines $A \leq pB$ by a polynomial-time transformation preserving membership. Compute that transformation and run B's algorithm to solve A. Therefore the target B is at least as hard under this reduction. The reduction does not force equal complexity classes, a reverse reduction or an algorithm for B from A alone.

Why each choice is right or wrong

A - Incorrect: This reverses the hardness direction implied by reducing A to B.

B - Incorrect: A one-way reduction does not place both languages in the same complexity class.

C - Correct: The polynomial mapping followed by a fast B solver gives a fast A solver.

D - Incorrect: A solver for A alone does not invert the reduction to provide a B solver.

Primary note proof

Study-note section AUT35

> $A \leq pB$ means f is computable in polynomial time and $x \in A$ iff $f(x) \in B$. Solving B then solves A, so B is at least as hard under this reduction. If $B \in P$ then $A \in P$; if A is NP-hard then B is NP-hard. $A \in P$ does not force $B \in P$, and the reduction alone does not place both problems in the same class. Polynomial reductions compose.

Question sources

Automata & Complexity Theory review questions · Question 48

Question 149 | NS0214

For standard finite encoded instances, how is the exact optimization task of finding a shortest tour visiting every city once and returning to the start classified?

A. P - solvable efficiently

B. PSPACE-complete

C. Not a formal computational problem

D. NP-hard; this classification does not establish an exponential-time lower bound.

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer D: NP-hard; this classification does not establish an exponential-time lower bound.

Finding the shortest tour is an optimization problem and is NP-hard. The related question whether a tour of cost at most a given bound exists is a decision problem and is NP-complete. These statements do not prove that every exact algorithm must take exponential time or resolve P versus NP. The option retains the established hardness classification without overstating a lower bound.

Why each choice is right or wrong

A - Incorrect: A general polynomial-time exact algorithm is not the classification established in the note.

B - Incorrect: PSPACE-complete is not the supplied classification of this optimization task.

C - Incorrect: The finite encoded task is a well-defined computational optimization problem.

D - Correct: Shortest-tour optimization is NP-hard; hardness alone gives no exponential lower bound.

Primary note proof

Study-note section AUT36

> TSP decision asks whether a tour of cost at most B exists; it is NP-complete for standard finite encoded inputs. Finding the shortest tour is optimization and NP-hard; call it NP-complete only after specifying a decision formulation.

Question sources

Automata & Complexity Theory review questions · Question 49

Question 150 | NS0215

Which statement about P vs NP is CORRECT?

A. It has been proven that P ≠ NP

B. It is an UNSOLVED open question - one of the most important problems in computer science

C. It has been proven that P = NP

D. The question is irrelevant since all real-world problems are in P

Hint

Separate resource bounds, certificate verification and reduction requirements. Use proved containments without assuming unresolved class separations.

Answer B: P versus NP remains open.

AUT34 explicitly says P=NP remains open, and AUT39 lists the separation among unresolved questions. Therefore neither equality nor inequality is proved in this source. The inclusion P⊆NP is known, but does not decide equality. Real-world problem encodings and resource requirements do not make the theoretical question irrelevant.

Why each choice is right or wrong

A - Incorrect: The note states that the separation is unresolved, not proved.

B - Correct: Whether P and NP are equal is an open question.

C - Incorrect: The note does not establish equality and explicitly calls it unresolved.

D - Incorrect: The note gives hard and undecidable problems, refuting the claim that all real-world problems fall in P.

Primary note proof

Study-note section AUT34

> Complexity classes here contain decision languages. P consists of deterministic polynomial-time decidable languages; NP consists of nondeterministic polynomial-time decidable languages, equivalently polynomially verifiable yes-instances with polynomial-size certificates. A verifier must accept some certificate exactly for yes-inputs; no certificate may falsely establish a no-instance. NP means nondeterministic polynomial, not non-polynomial.

Study-note section AUT34

> P⊆NP because a verifier can ignore its certificate and run the P algorithm. P=NP remains open; NP does not mean definitely hard or definitely outside P.

Study-note section AUT39

> Use L⊆NL⊆P⊆NP⊆PSPACE⊆EXPTIME⊆EXPSPACE. Here L/NL use logarithmic workspace; EXPTIME is ⋃kTIME(2^(n^k)), an upper-bound class that also contains P, not a statement that every member requires exponential time.

Study-note section AUT39

P versus NP, NP versus PSPACE and PSPACE versus EXPTIME remain unresolved. coNP contains complements of NP languages; P⊆NP∩coNP, and NP=coNP is open.

Question sources

Automata & Complexity Theory review questions · Question 50

Question 151 | NS0287

Which automaton is powerful enough to recognize the language L = {$a^n b^n c^n$ | n ≥ 1}?

A. DFA

B. Linear Bounded Automaton (LBA)

C. NFA

D. Pushdown Automaton (PDA)

Hint

Respect the quantifier order and length constraints. A valid nonmembership argument must address every allowed decomposition, not one convenient split.

Answer B: A linear bounded automaton.

An LBA checks block order then repeatedly marks one a, one b and one c, rejecting unmatched symbols. The language is not context-free, ruling out a PDA and therefore finite automata too.

Why each choice is right or wrong

A - Incorrect: A DFA recognizes only regular languages, too weak for this non-CFL language.

B - Correct: An LBA can match the three counts by marking symbols within the bounded input region.

C - Incorrect: NFA power equals DFA power and cannot reach this context-sensitive non-CFL language.

D - Incorrect: A PDA recognizes CFLs, whereas this equal-three-count language is not context-free.

Primary note proof

Study-note section AUT25

A context-sensitive rule αAβ→αγβ replaces A in context, with γ nonempty. Equivalently, noncontracting grammar presentations preserve/increase total length, with the isolated ε exception. The condition is on the complete sides, not an unrelated comparison |β|≥|α|.

Study-note section AUT25

A linear bounded automaton (LBA) is a TM with O(n) tape cells on length-n input; the standard CSL characterization uses nondeterministic LBAs. For $a^n b^n c^n$ first verify a*b*c* order, repeatedly mark one unmarked a, b and c, and reject missing/excess symbols.

Study-note section AUT25

Thus every CSL is decidable.

Study-note section AUT24

If L is context-free, some p ensures each sufficiently long $s \in L$ has $s=uvxyz$ with $|vxy| \le p$, $|vy| \ge 1$ and $uv^ixy^iz \in L$ for every $i \ge 0$. The two pieces v and y are pumped together; either may be empty, not both. A non-CFL proof must defeat every allowed decomposition.

Study-note section AUT24

For $s=a^pb^pc^p$, the window vxy cannot contain both an a and a c. Pumping affects at most two blocks and at least one count. Use i=0 or 2: the unaffected third block retains p while another count changes, or symbol order breaks. Hence $a^nb^nc^n$ is not CFL.

Question sources

Computer Science Exit Exam · 2018 · Question 4

Question 152 | NS0309

Which of the following statements is TRUE?

A. Every regular language is context-free.

B. Every context-free language is regular.

C. Every context-sensitive language is regular.

D. Every recursively enumerable language is recursive

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Every regular language is context-free.

AUT05 gives REG⊊CFL⊊CSL⊊DECIDABLE⊊RE, immediately establishing A. Reverse inclusions for CFL or CSL into REG fail. AUT29 explains that RE strictly exceeds decidable/recursive languages, so not every RE language is recursive. Strictness is essential to rejecting the three converse-style claims.

Why each choice is right or wrong

A - Correct: REG is a proper subfamily of CFL, so every regular language is context-free.

B - Incorrect: Matched equal counts supply CFLs that are not regular.

C - Incorrect: CSL includes nonregular and even non-CFL languages.

D - Incorrect: Some Turing-recognizable languages, such as HALT, are not recursive.

Primary note proof

Study-note section AUT05

> Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

> Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Question sources

Computer Science Exit Exam · 2018 · Question 30

Question 153 | NS0322

In the CNF-first route for converting a CFG into Greibach normal form, which initial normalization is performed before the later ordering, substitution and left-recursion work?

A. Convert the grammar into CNF

B. If any production rule in the grammar is not in GNF form, convert it.

C. Convert the grammar into GNF

D. If the grammar exists left recursion, eliminate it.

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer A: Convert the grammar into CNF

The specified route begins by putting the grammar into Chomsky normal form. Later ordering and substitution arrange terminal-leading productions and eliminate left recursion as required for Greibach form. CNF is an intermediate normalization in this particular route, not a universal prerequisite for every GNF construction. The other conversion choices state the goal broadly rather than identify this route's initial normalization.

Why each choice is right or wrong

A - Correct: CNF conversion is the initial normalization in the selected route.

B - Incorrect: Converting arbitrary offending rules does not identify the route's initial normalization.

C - Incorrect: GNF conversion is the overall goal, not this preliminary operation.

D - Incorrect: Left recursion must be handled later in this chosen route.

Primary note proof

Study-note section AUT20

> Some algorithms first convert to CNF; this is a method choice, not a universally mandatory first step.

Question sources

Computer Science Exit Exam · 2018 · Question 47

Question 154 | NS0336

A recognizer for a language L halts and accepts every member, but may reject or run forever on nonmembers. L is known to be undecidable. Which listed grammar family can nevertheless generate exactly L?

A. Context-free

B. Context-sensitive

C. Unrestricted (Type 0)

D. Regular

Hint

Check whether the required procedure must halt on every input, and distinguish solving a task from recognizing its yes-instances.

Answer C: Unrestricted (Type 0)

The recognizer accepts every member, so L is Turing-recognizable even though no decider exists for it. Type 0 unrestricted grammars generate exactly the Turing-recognizable languages and can therefore generate L. Regular, context-free and context-sensitive languages are all decidable under the stated hierarchy, so none of those grammar families can generate this known undecidable language. The distinction is recognition versus guaranteed decision, not running speed.

Why each choice is right or wrong

A - Incorrect: Every context-free language is decidable, whereas the given L is not.

B - Incorrect: Every context-sensitive language is decidable, excluding the stated L.

C - Correct: Type 0 grammars correspond to Turing recognizers, including recognizable undecidable languages.

D - Incorrect: Every regular language is decidable, whereas the given L is not.

Primary note proof

Study-note section AUT29

> A decider halts on every input and accepts exactly its language; these languages are recursive/decidable. A recognizer must halt and accept every member, but on a nonmember may halt-reject or loop. Its languages are recursively enumerable (RE), Turing-recognizable or semidecidable. Looping is nonacceptance, not a rejecting halt. An undecidable language may still have a recognizer.

Study-note section AUT29

> Every decidable language is RE, and the containment is strict.

Study-note section AUT05

Type 3 regular grammars correspond to finite automata; type 2 CFGs to nondeterministic PDAs; type 1 context-sensitive/noncontracting grammars to nondeterministic LBAs; type 0 unrestricted grammars to Turing recognizers.

Study-note section AUT05

Use REG ⊊ CFL ⊊ CSL ⊊ DECIDABLE ⊊ RE, under the usual convention allowing S→ε with S absent from right sides when needed.

Question sources

Computer Science Exit Exam · 2018 · Question 66

Question 155 | NS0353

ou have given that Grammar G1 −({S, A, B}, {a, b}, S, {S → AB, A → a, B → b}).
Which one is terminal symbol in the give?

A. a,b

B. S

C. S → AB, A → a, B → b

D. S,A,B

Hint

Distinguish symbols, finite strings, their lengths and sets of strings. Apply the operation to its stated operands, including empty cases.

Answer A: a and b are the terminal symbols.

AUT04 separates the variable set from the terminal alphabet in G. The stem explicitly declares {S,A,B} as variables and {a,b} as terminals. The production rules are rewrites, not symbols, and uppercase S is the start variable. Therefore a,b is the terminal-symbol choice without needing any derivation calculation.

Why each choice is right or wrong

A - Correct: These are exactly the two symbols in the supplied terminal alphabet {a,b}.

B - Incorrect: S is the start variable rather than a terminal.

C - Incorrect: These are production rules, not terminal symbols.

D - Incorrect: S,A,B is the declared variable set rather than the terminal alphabet.

Primary note proof

Study-note section AUT04

> Write G=(V,Σ,P,S): finite variables V, terminals Σ disjoint from V, productions P and start variable S∈V. A production replaces its left side by its right side. The language is L(G)={w∈Σ*:S⇒*w}. A recognizer processes a word and accepts exactly the language members. Grammars generate languages; automata recognize languages, not grammars.

Study-note section AUT04

> Two grammars are equivalent when their generated languages are equal, regardless of rule counts or variable names. One common word, equal alphabets or one-way inclusion is insufficient.

Question sources

Computer Science Exit Exam · 2018 · Question 90

Question 156 | NS0356

The language $L = \{ww^R \mid w \in \{a, b\}^*\}$ (even-length palindromes) is:

A. Regular

B. Non-deterministic CFL but not deterministic

C. Deterministic CFL

D. Context-sensitive but not context-free

Hint

Compare production restrictions, memory access and language-family containments. Distinguish an exact characterization from an included subclass.

Answer B: A nondeterministic CFL that is not deterministic context-free.

AUT22 gives the NPDA for binary ww^R by guessing the midpoint and states that without a marker it is not deterministic context-free. The alphabet {a,b} is explicit here, excluding the unary regular exception. A visible center marker would allow a DPDA, but no such marker occurs in the given language.

Why each choice is right or wrong

A - Incorrect: Binary unmarked even palindromes are not regular; the unary exception does not apply.

B - Correct: An NPDA guesses the midpoint, while the unmarked binary language is not deterministic context-free.

C - Incorrect: Deterministic recognition requires a visible midpoint marker for the contrasting construction in AUT22.

D - Incorrect: The language is context-free, so denying context-freeness is false even though it is also CSL.

Primary note proof

Study-note section AUT22

> For even palindromes ww^R over {a,b}, push the first half, nondeterministically guess its midpoint, then pop matching symbols. No center marker makes that language context-free but not deterministic context-free. With a marker wcw^R the switch point is visible and a DPDA suffices. Unary even palindromes are merely even-length words and regular. A PDA cannot compare two arbitrary halves in the same forward order: $\{ww : w \in \{a,b\}^*\}$ is the copy language, not the palindrome language.

Question sources

Computer Science Exit Exam · 2018 · Question 94

Compiler Design

Questions & Worked Answers

200 questions

Question 1 | Q01406

Which grammar gives multiple parse trees for the same string?

A. Unambiguous

B. Regular

C. Ambiguous

D. All of the above

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: An ambiguous grammar.

COMP10 defines ambiguity by the existence of one terminal string with two different parse trees. This is exactly the property asked for. An unambiguous grammar excludes it. Being regular describes a grammar class rather than guaranteeing multiple trees, so it does not make all the listed descriptions correct.

Why each choice is right or wrong

A - Incorrect: Unambiguous means each generated string has a unique parse tree, directly contradicting the property.

B - Incorrect: Regularity restricts grammar form; it does not require multiple parse trees for a string.

C - Correct: Ambiguity is precisely the existence of multiple parse trees for the same terminal string.

D - Incorrect: Unambiguous grammars do not have the stated property, so all cannot be correct.

Primary note proof

Study-note section COMP10

> A grammar is ambiguous if at least one terminal string has two different parse trees, equivalently two leftmost derivations. Merely changing derivation order on the same tree does not establish ambiguity. E→E+E|E*E|id allows different trees for id+id*id.

Study-note section COMP10

> Use E→E+T|T, T→T*F|F, F→(E)|id to give * higher precedence and both operators left associativity. For subtraction, a−b−c then groups (a−b)−c. The dangling-else grammar permits

else to attach to either nested if; a nearest-unmatched-if rule or grammar rewrite resolves it. Taking a leftmost derivation by itself does not resolve ambiguous interpretation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 6-16

Grammars
• Every programming language has grammar rules
• Parsers or syntax analyzers are generated for a particular grammar
• CFG are used for syntax specification of programming languages
Context Free Grammar (CFG)
• Is denoted as G = (N, T , P, S)
• N : finite set of non-terminals
• T : finite set of terminals
• S ∈ N: The start symbol
• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Compiler Design module | PDF page 9 | lines 22-27

Derivation Trees
• Derivations can be displayed as trees
• Internal nodes of the tree are all non-terminals
• Leaves are all terminals
• The yield of a derivation tree is the list of the labels of all the leaves read from left to right.

Question sources

Compiler Design review questions · Question 14

Question 2 | Q01416

Leaf nodes in a parse tree indicate?

A. sub-terminals

B. half-terminals

C. non-terminals

D. terminals

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: terminals

PDF p. 9 explicitly states that internal nodes are non-terminals and leaves are terminals. Thus the leaf labels in this module's derivation trees are terminals, not the grammar variables expanded at internal nodes.

Why each choice is right or wrong

A - Incorrect: Sub-terminals is not a category in the module's CFG or tree definitions (p. 9).

B - Incorrect: Half-terminals is not a category in those definitions (p. 9).

C - Incorrect: Non-terminals label internal nodes, not leaves (p. 9).

D - Correct: The module expressly assigns terminals to leaves (p. 9).

Primary note proof

Study-note section COMP09

> A context-free grammar G=(N,T,P,S) has nonterminals N, terminals T, productions P and start S. A production A→α rewrites one nonterminal. A sentential form is any string of terminals/nonterminals derivable from S; a sentence contains only terminals. A leftmost derivation rewrites the leftmost remaining nonterminal; a rightmost one rewrites the rightmost.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input. An AST omits grammar scaffolding and punctuation; expression operands are leaves and operators internal nodes. The parser can build either representation, depending on implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 22-27

> Derivation Trees
> • Derivations can be displayed as trees
> • Internal nodes of the tree are all non-terminals
> • Leaves are all terminals
> • The yield of a derivation tree is the list of the labels of all the leaves read from left to right.

Question sources

Compiler Design review questions · Question 24

Question 3 | Q00047

Which one of the following is not the task of lexical analyzer?

A. Group characters into lexemes

B. Output a sequence of tokens

C. Read the input characters

D. Build syntax tree

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Build syntax tree

Scanning reads characters and represents matched lexemes as tokens (PDF pp. 5, 8). Syntax analysis then groups those tokens and produces a parse tree. Building that tree is therefore the non-lexical task.

Why each choice is right or wrong

A - Incorrect: Grouping characters into matched lexemes belongs to scanning (pp. 5, 8).

B - Incorrect: A sequence of tokens is the scanner's stated output (p. 8).

C - Incorrect: Reading the stream of source characters is explicitly lexical analysis.

D - Correct: Constructing the syntactic tree belongs to syntax analysis.

Primary note proof

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

> 1. Lexical / Linear Analysis (scanning)
> - Scans the source code as a stream of characters
> - Represent lexemes in the form of tokens as:
> <token-name, attribute-value>
> - Token

• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 47

Question 4 | Q00054

Which statement is false about a compiler parser that constructs a parse tree and supports syntax-error diagnosis and recovery?

A. Parser builds the parse tree

B. Parser attempts to correct or recover few errors

C. Parser cannot perform context free syntax analysis

D. Parser produces appropriate error messages

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: Parser cannot perform context free syntax analysis

A parser analyzes the token stream against the context-free syntax and builds its structural representation. In a compiler with syntax diagnostics and recovery, it can report an unexpected token and try to resume at a useful boundary. Therefore the claim that it cannot perform context-free syntax analysis reverses its defining role. Building a parse tree, reporting syntax errors and attempting the specified recovery behavior are compatible responsibilities.

Why each choice is right or wrong

A - Incorrect: Building a parse tree represents the grammar structure recognized by syntax analysis.

B - Incorrect: Error recovery may skip to a synchronization token or make a limited local repair.

C - Correct: Context-free syntax analysis is the parser's defining task, so denying it is false.

D - Incorrect: Syntax diagnostics identify unexpected or missing tokens with useful source locations.

Primary note proof

Study-note section COMP09

> A context-free grammar G=(N,T,P,S) has nonterminals N, terminals T, productions P and start S. A production A→α rewrites one nonterminal.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Study-note section COMP39

> Panic-mode recovery skips input until a synchronizing token such as ; or } permits parsing to resume. Phrase-level recovery makes a local insertion/deletion. Report the location and cause while avoiding cascades of misleading diagnostics. Lexical, syntax and semantic phases each detect their own errors; error handling is a shared service.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 6-16

> Grammars
> • Every programming language has grammar rules
> • Parsers or syntax analyzers are generated for a particular grammar
> • CFG are used for syntax specification of programming languages
> Context Free Grammar (CFG)
> • Is denoted as G = (N, T , P, S)
> • N : finite set of non-terminals
> • T : finite set of terminals
> • S ∈ N: The start symbol

• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Question sources

MOE Computer Science Exit Exam · 2016 · Question 54

Question 5 | Q00058

For an infix source expression targeting a postfix stack evaluator, which notation conversion prepares it for evaluation?

A. Convert infix notation to postfix notation

B. Convert postfix notation to prefix notation

C. Evaluate infix notation to prefix notation

D. Convert postfix to infix notation

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer A: Convert infix notation to postfix notation

A postfix evaluator reads operands before the operator that combines them. Converting an infix expression to postfix therefore supplies the required input order for the specified stack target. For example, the tree for a−4+c yields a 4 − c + in postorder. This is a choice of representation for this evaluator, rather than a mandatory strategy for every compiler.

Why each choice is right or wrong

A - Correct: Infix-to-postfix supplies the required target order.

B - Incorrect: Neither source nor target matches this direction.

C - Incorrect: Prefix is not the specified target.

D - Incorrect: This reverses the specified translation direction.

Primary note proof

Study-note section COMP25

A syntax-directed translation scheme embeds executable actions in productions; the actions run while translating a source program, not while generating the compiler. Attributes may compute types, values, AST pointers or generated code. The scheme defines an input-to-output translation, not merely an input language.

Study-note section COMP25

Create an identifier leaf with mkleaf(id,entry), a number with mkleaf(num,value), and an operator node with mknode(op,left,right). For a−4+c, construct +(−(a,4),c). Postorder traversal prints postfix: a 4 − c +. A stack evaluator pushes operands and pops right then left operands for an operator. Infix-to-postfix is useful for stack machines but not mandatory for all compilers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 17-25

Syntax Directed Definition- Example
• Production Semantic Rules
• L → E return print(E.val)
• E → E1 + T E.val = E1.val + T.val
• E → T E.val = T.val
• T → T1 * F T.val = T1.val * F.val
• T → F T.val = F.val
• T → (E) F.val = E.val
• F → digit F.val = digit.lexval

Compiler Design module | PDF page 16 | lines 2-8

Three Address Code
• Is a generic form and can be implemented as:
- Quadruples
- Triples
- Indirect Triples
- Tree
- DAG

Question sources

MOE Computer Science Exit Exam · 2016 · Question 58

Question 6 | Q00335

Which named transformation directly replaces an expression whose operands are compile-time constants with its computed value?

A. Pee hole optimization

B. Constant folding

C. Invariant computation

D. Code hoisting

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer B: Constant folding

Constant folding evaluates an expression from operands known at compilation and substitutes its value, such as replacing 3*4 with 12. This removes that expression's run-time arithmetic. A peephole pass describes a small inspection region and may perform several transformations; it is not the particular evaluation rule. Loop-invariant computation can remain unknown until execution, and hoisting moves safe invariant work rather than necessarily computing a compile-time value.

Why each choice is right or wrong

A - Incorrect: Peephole identifies a short-window optimization method, not specifically constant-expression evaluation.

B - Correct: Constant folding computes a known constant expression during compilation.

C - Incorrect: An invariant may depend on a run-time input even though it stays unchanged across a loop.

D - Incorrect: Hoisting moves a computation to a preheader; it need not evaluate it at compilation.

Primary note proof

Study-note section COMP30

> Constant folding evaluates constant expressions during compilation: 3*4→12.

Study-note section COMP32

> A loop-invariant expression has unchanged inputs during a loop. Hoist it to a preheader only when evaluation is safe on the paths where it will newly run.

Study-note section COMP31

> Peephole optimization examines short instruction windows.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 18 | lines 23-34

Peephole Optimization
• Transforming to optimal sequence of instructions
Common Techniques:
• Elimination of redundant loads and stores
- Eg.
- r2 = r1 + 5
- I = r2
- r3 = I
- r4 = r3 * 3

• Constant folding
- Eg.
- R2 = 3 * 2

Compiler Design module | PDF page 18 | lines 2-10

Techniques
• Common sub-expression elimination
- Repeated appearance computed previously

• Strength reduction
- Replacement of expensive expressions with simple ones

• Code movement
- Moving a block of code outside a loop

• Dead code elimination
- Eliminated code statements that are either never executed or unreachable

Question sources

Computer Science model exam · Question 66

University of Gondar model exam · Question 66

Question 7 | Q00337

Which listed deterministic parser families can evaluate synthesized attributes for suitable grammars and semantic actions? Select all that apply.

A. Ambiguous grammar

B. LR grammar

C. LL grammar

D. None

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer B, C: LR grammar; LL grammar

Synthesized attributes flow from child nodes to their parent. LR parsers naturally compute these during reduction, after the right-hand-side children have been recognized. Suitable LL implementations can also compute them after their recursive child calls return. Ambiguity is a property of grammar interpretation, not a deterministic parser family or attribute-evaluation method. Thus both LR and LL are valid families for this task.

Why each choice is right or wrong

A - Incorrect: Ambiguity is not a deterministic parser family or evaluation mechanism.

B - Correct: LR reduction actions can compute parent values from children.

C - Correct: Suitable LL actions can compute values after child processing.

D - Incorrect: Both LR and LL provide supported implementations.

Primary note proof

Study-note section COMP24

> A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions. An annotated parse tree displays computed attribute values; it is not itself an inherited or synthesized attribute. Synthesized attributes flow from children to a parent; inherited attributes carry context from parent or siblings.

Study-note section COMP24

> For E→E1+T, E.val=E1.val+T.val synthesizes an expression value. A declaration's type environment passed into its body is inherited. S-attributed definitions use only synthesized attributes and fit LR reduction-time evaluation naturally; suitable LL parsers can also compute them. L-attributed definitions restrict dependencies so a left-to-right traversal can evaluate inherited context. Attribute dependency cycles can prevent evaluation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 4-16

> Syntax Directed Translation
> • Attaching actions to the grammar rules(productions).
> • Actions are executed during the compilation
> ● Not during the generation of the compiler
> • Actions are executed according to the parsing mechanism.
> Syntax Directed Definitions
> • Is a generalization of a context free grammar
> • Is a CFG with attributes and rules
> • Attributes are associated with grammar symbols and rules with productions
> • Attributes may be:
> ● Numbers
> ● Types
> ● Strings etc

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science model exam · Question 68

University of Gondar model exam · Question 68

Question 8 | Q00338

Identifiers consist of one Letter followed by zero or more Letters or Digits. Which expressions denote exactly that language? Select all that apply. Here + means one-or-more, | alternation, and _ a literal underscore.

A. Letter(Letter+|Digit*)*

B. Letter(Letter|Digit)+

C. Letter(Letter|Digit)*

D. Letter(_Digit)*

Hint

Follow the representation at each construction step and apply the pattern or state-set rules consistently, including empty cases.

Answer A, C: Letter(Letter+|Digit*)*; Letter(Letter|Digit)*

C directly expresses a leading letter followed by any sequence of letters and digits. A is equivalent: its outer star may take zero blocks, and repeated letter or digit blocks can generate every such suffix. B excludes a one-letter identifier because its suffix uses plus. D permits literal underscores and restricted digit blocks, so it does not denote the specified identifier language.

Why each choice is right or wrong

A - Correct: Its repeated blocks generate every allowed suffix, including the empty suffix.

B - Incorrect: The trailing plus excludes a one-letter identifier.

C - Correct: This directly states the required pattern.

D - Incorrect: Its literal underscores violate the specified alphabet and its restricted suffix misses identifiers.

Primary note proof

Study-note section COMP05

> Regular expressions use concatenation, alternation r|s, Kleene star r* for zero or more, and r+ for one or more. In a simplified alphabet, identifier = Letter(Letter|Digit)*; it permits a single letter and forbids a leading digit.

Study-note section COMP05

> Letter(Letter+|Digit*)* denotes the same language as Letter(Letter|Digit)* under ordinary regular-expression semantics: repeated blocks can contain any trailing letter/digit sequence. These two regular expressions are equivalent. Regular patterns handle ordinary tokens; arbitrary balanced nesting generally needs grammar/stack structure.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 10-18

> Tokens, Patterns, and Lexemes
> • Token
> ● A string of characters which logically belong together
> ● Classes of similar lexemes
> l identifier, keywords, constants etc.
> • Pattern
> ● A rule which describes a token
> • Lexeme
> ● The sequence of characters matched by a pattern to form the token

Compiler Design module | PDF page 8 | lines 19-24

> • Classes of Tokens
> ● Identifiers: names chosen by the programmer
> ● Keywords: names already in the programming language
> ● Separators: punctuation characters
> ● Operators: symbols that operate on arguments and produce results
> ● Literals: numeric, textual literals

Question sources

Computer Science model exam · Question 69

University of Gondar model exam · Question 69

Question 9 | Q00339

What is the process of finding a parse tree for a string of tokens called?

A. Recognizing

B. Parsing

C. Tokenizing

D. Analyzing

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Parsing

Parsing is explicitly the process of constructing a parse tree for a sentence generated by a grammar (PDF p. 10). That is the operation named in the question; token production precedes it (p. 8).

Why each choice is right or wrong

A - Incorrect: The module does not use Recognizing as the name of this construction (p. 10).

B - Correct: Parsing matches the express definition (p. 10).

C - Incorrect: Tokenizing produces the token stream, rather than its parse tree (p. 8).

D - Incorrect: Analyzing is a broad description; the defined specific process is Parsing (p. 10).

Primary note proof

Study-note section COMP09

> A context-free grammar G=(N,T,P,S) has nonterminals N, terminals T, productions P and start S. A production A→α rewrites one nonterminal. A sentential form is any string of terminals/nonterminals derivable from S; a sentence contains only terminals. A leftmost derivation rewrites the leftmost remaining nonterminal; a rightmost one rewrites the rightmost.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input. An AST omits grammar scaffolding and punctuation; expression operands are leaves and operators internal nodes. The parser can build either representation, depending on implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

> Parsing
> • Process of constructing parse tree for a sentence generated by a given grammar.

- 2 types of parsers
  - Top down parsing (predictive parsers)
- LL(1)
  - Bottom up parsing (SR parsers)
- LR(1)

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
- The first phase of a compiler
- The input is a high level language program
- The output is a sequence of tokens
- Strips off blanks, tabs, newlines, and comments from the source program
- Keeps track of line numbers

Question sources

Computer Science model exam · Question 70

University of Gondar model exam · Question 70

Question 10 | Q00340

Which compiler phase groups source characters into lexemes and classifies them as identifier, keyword, literal or operator tokens?

A. Interpretation analysis

B. Lexical analysis

C. Syntax analysis

D. General syntax analysis

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Lexical analysis

Lexical analysis groups the source character sequence into lexemes and emits token categories with attributes. Syntax analysis instead combines already recognized tokens into larger grammatical structures such as expressions and statements. Specifying characters and token categories resolves the original mixture of lexical classification and parsing terminology. The generic interpretation and syntax labels do not describe the requested character-to-token phase.

Why each choice is right or wrong

A - Incorrect: Interpretation executes a representation rather than classifying lexemes.

B - Correct: Lexical analysis turns character sequences into tokens.

C - Incorrect: Syntax analysis groups tokens into grammatical structures.

D - Incorrect: This syntax label concerns parsing rather than lexical classification.

Primary note proof

Study-note section COMP41

> Grouping characters into token classes is lexical analysis; grouping tokens into expressions and statements is syntax analysis.

Study-note section COMP41

> Trees store hierarchical constructs, while a symbol table stores declaration metadata.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

> 1. Lexical / Linear Analysis (scanning)
> ● Scans the source code as a stream of characters
> ● Represent lexemes in the form of tokens as:
> <token-name, attribute-value>
> ● Token
> • smallest meaningful element that a compiler understands.
> ● Eg.
> • Identifiers, Keywords, Literals, Operators and Special symbols.
> ● Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 5 | lines 12-18

> 2. Syntax / Hierarchical Analysis - Parsing
> • Tokens are grouped hierarchically into nested collections with collective meaning.
> • The result is generally a parse tree.
> • expressions, statements, declarations etc... are identified by using the results of lexical analysis.
> • Most syntactic errors in the source program are caught in this phase.
> • Syntactic rules of the source language are given via a Grammar.

Question sources

Computer Science model exam · Question 71

University of Gondar model exam · Question 71

Question 11 | Q00532

For a register-based target compiler, which choice most completely lists general code-generator tasks?

A. Instruction selection

B. Register allocation and assignment

C. Instruction ordering

D. All

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: All

COMP34 separates selecting target operations, deciding storage in registers, assigning physical registers, and scheduling instructions. The three concrete choices name these standard code-generation tasks. Since the question offers an aggregate covering them, that aggregate is the complete answer; each individual choice supplies only part of the requested task list.

Why each choice is right or wrong

A - Incorrect: True as one component, but incomplete as the requested full selection. Instruction selection chooses which target instructions implement the intermediate operations.

B - Incorrect: True as one component, but incomplete as the requested full selection. Register allocation and physical assignment decide where values reside and which registers hold them.

C - Incorrect: True as one component, but incomplete as the requested full selection. Instruction ordering schedules computations while preserving data and control dependencies.

D - Correct: All three listed activities belong to the code-generator task set described in COMP34.

Primary note proof

Study-note section COMP34

> Code generation lowers checked IR to a target form. Instruction selection chooses target operations; register allocation chooses where values reside; scheduling orders instructions while respecting dependencies; storage layout assigns offsets and follows ABI requirements. Register assignment names actual physical registers after allocation decisions.

Study-note section COMP34

> For t=a+b, a simple target could emit LOAD R1,a; LOAD R2,b; ADD R3,R1,R2; STORE t,R3. Other targets may combine loads with arithmetic or keep t in a register. Machine idioms, register availability and instruction order affect execution time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Compiler Design module | PDF page 18 | lines 11-22

Register Allocation
• Registers hold values
• Example
- a = c + d
- e = a + b
- f = e - 1

• With the assumption that a and e die after use
• Temporary a can be reused after e=a+b, same wz a
• Can allocate a,e and f all to one register(r1)
- r1 = r2 + r3
- r1 = r1 + r4
- r1 = r1 - 1

Question sources

Computer Science model exam · 2015 · Question 1

Question 12 | Q00537

What is the term for an actual occurrence of a character sequence in source text that matches a token pattern, such as total or 42?

A. Pattern

B. Lexemes

C. Tokens

D. All

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Lexemes

A lexeme is the actual character sequence occurring in source text. A token is the category and optional attribute emitted for that occurrence, while a pattern defines which spellings are members of the category. For example, total is a lexeme classified as an identifier token. These are related but distinct objects, so the aggregate choice cannot name the concrete occurrence.

Why each choice is right or wrong

A - Incorrect: A pattern describes possible spellings.

B - Correct: Lexeme names the actual matched source sequence.

C - Incorrect: Token names the category and optional attribute.

D - Incorrect: The three concepts are distinct, not synonymous.

Primary note proof

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP41

Grouping characters into token classes is lexical analysis; grouping tokens into expressions and statements is syntax analysis.

Study-note section COMP41

Trees store hierarchical constructs, while a symbol table stores declaration metadata.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

1. Lexical / Linear Analysis (scanning)
- Scans the source code as a stream of characters
- Represent lexemes in the form of tokens as:
<token-name, attribute-value>
- Token
• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 8 | lines 10-18

Tokens, Patterns, and Lexemes
• Token
- A string of characters which logically belong together
- Classes of similar lexemes
l identifier, keywords, constants etc.
• Pattern

- A rule which describes a token
- Lexeme
- The sequence of characters matched by a pattern to form the token

Question sources

Computer Science model exam · 2015 · Question 6

Question 13 | Q00561

Which one of the following are the specific character strings that make up a token?

A. Token

B. Lexemes

C. Lexical units

D. Pattern

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Lexemes

A lexeme is the sequence of characters matched by a pattern to form a token (PDF p. 8). A token instead denotes a class of similar lexemes, and a pattern specifies the rule. “Specific character strings” therefore identifies lexemes.

Why each choice is right or wrong

A - Incorrect: Token names the class in the module’s token/lexeme distinction (p. 8).

B - Correct: Lexemes are the actual matched sequences (p. 8).

C - Incorrect: Lexical units is not the defined name for specific strings in this passage (p. 8).

D - Incorrect: A pattern is the matching rule, not its particular match (p. 8).

Primary note proof

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive

or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 10-18

Tokens, Patterns, and Lexemes
• Token
- A string of characters which logically belong together
- Classes of similar lexemes

l identifier, keywords, constants etc.
• Pattern
- A rule which describes a token

• Lexeme
- The sequence of characters matched by a pattern to form the token

Question sources

Computer Science model exam · 2015 · Question 30

Question 14 | Q00578

Which one of the following are the program properties that can be checked at run time?

A. Static checking

B. Dynamic semantics

C. Other Static Checks

D. None

Hint

Ask whether the required information is a declaration’s identity and attributes, program structure or the ordered source tokens.

Answer B: Dynamic semantics

Semantic checks are static when done during compilation and dynamic when done during run-time (PDF p. 14). Interpreting “Dynamic semantics” as the option’s name for run-time semantic checking, B fits; static checking and other static checks occur during compilation.

Why each choice is right or wrong

A - Incorrect: Static checking is done at compilation, not run-time (p. 14).

B - Correct: B corresponds to the module’s run-time dynamic semantic checks (p. 14).

C - Incorrect: Other static checks retain the compile-time timing (p. 14).

D - Incorrect: None does not fit because the dynamic check category is present (p. 14).

Primary note proof

Study-note section COMP21

Static checks occur before execution: type compatibility, duplicate declarations within a restricted scope, valid names, argument counts, return compatibility and control-flow placement. A break is valid only in constructs allowed by the language; in C/C++ it can exit a loop or switch. A compiler may annotate an AST or create typed IR; there is no mandatory uniquely named output tree.

Study-note section COMP21

Dynamic checks depend on runtime values, such as some bounds and null checks. Static analysis can prove some accesses safe or faulty; “all memory errors cannot be statically checked” is too strong. Unreachable code is not universally a semantic error: some languages reject it in specified cases, others merely warn or optimize it away.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 14 | lines 2-15

• Type Errors:
- Improper or inconsistent operations during program execution
• Type-safety:
- Absence of type errors

Type Checking
• Semantic checks to enforce the type safety of the program
• Semantic Checks
- Static - done during compilation
- Dynamic - done during run-time
• Examples
- Unary and binary operators
- Number and type of arguments
- Return statement with return type
- Compatible assignment

Compiler Design module | PDF page 14 | lines 16-23

Static Checking
• The compiler must check the semantic conventions of the source language
• Static Checking: ensures that certain kind of errors are detected and reported
• Example
• Type Checks: incompatible operands
• Flow Control Check
• Uniqueness Check
• Name Related Check

Question sources

Computer Science model exam · 2015 · Question 47

Question 15 | Q00610

For a language with insignificant whitespace and comments, which choice most completely lists the scanner’s secondary tasks?

A. Stripping out blanks, tabs, new lines

B. Stripping out comments

C. Error reporting

D. All

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: All

COMP04 says scanners usually skip whitespace and comments while retaining source positions. COMP39 adds that lexical analysis detects its own errors and that error handling is shared. These establish all three listed secondary scanner duties under the ordinary language convention. Layout-sensitive languages are an explicit exception in COMP04, not evidence that scanners never perform these tasks.

Why each choice is right or wrong

A - Incorrect: True as one component, but incomplete as the requested full selection. Ordinary scanners skip blanks, tabs and newlines while retaining positions; layout-sensitive languages need special handling.

B - Incorrect: True as one component, but incomplete as the requested full selection. Comments are normally discarded instead of being delivered as ordinary parser tokens.

C - Incorrect: True as one component, but incomplete as the requested full selection. Lexical errors require reporting, with positions retained to locate the offending source text.

D - Correct: Together the choices cover normal whitespace removal, comment removal and lexical error reporting.

Primary note proof

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP39

> Report the location and cause while avoiding cascades of misleading diagnostics. Lexical, syntax and semantic phases each detect their own errors; error handling is a shared service.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Question sources

Computer Science model exam · 2015 · Question 79

Question 16 | Q00613

Which one of the following is the parse tree showing the values of attributes at each node?

A. Inherited attribute

B. Synthesized attribute

C. Annotated parse tree

D. A&B

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer C: An annotated parse tree.

COMP24 directly calls a parse tree displaying computed attribute values an annotated parse tree. Inherited and synthesized classify how attribute information flows; neither is the name of the entire displayed tree. Combining those attribute categories still does not name the representation described in the question.

Why each choice is right or wrong

A - Incorrect: An inherited attribute carries contextual information; it is not a parse tree displaying values.

B - Incorrect: A synthesized attribute carries a value upward, rather than naming the annotated tree.

C - Correct: This is the note's precise name for a parse tree showing attribute values at nodes.

D - Incorrect: Combining two attribute categories does not give the name of the displayed tree.

Primary note proof

Study-note section COMP24

> A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions. An annotated parse tree displays computed attribute values; it is not itself an inherited or synthesized attribute. Synthesized attributes flow from children to a parent; inherited attributes carry context from parent or siblings.

Study-note section COMP24

> For E→E1+T, E.val=E1.val+T.val synthesizes an expression value. A declaration's type environment passed into its body is inherited. S-attributed definitions use only synthesized attributes and fit LR reduction-time evaluation naturally; suitable LL parsers can also compute them. L-attributed definitions restrict dependencies so a left-to-right traversal can evaluate inherited context. Attribute dependency cycles can prevent evaluation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 4-16

> Syntax Directed Translation
> • Attaching actions to the grammar rules(productions).
> • Actions are executed during the compilation
> ● Not during the generation of the compiler
> • Actions are executed according to the parsing mechanism.
> Syntax Directed Definitions
> • Is a generalization of a context free grammar
> • Is a CFG with attributes and rules
> • Attributes are associated with grammar symbols and rules with productions
> • Attributes may be:
> ● Numbers
> ● Types
> ● Strings etc

Compiler Design module | PDF page 12 | lines 17-25

> Syntax Directed Definition- Example
> • Production Semantic Rules
> • L → E return print(E.val)
> • E → E1 + T E.val = E1.val + T.val
> • E → T E.val = T.val
> • T → T1 * F T.val = T1.val * F.val
> • T → F T.val = F.val

- T → (E) F.val = E.val
- F → digit F.val = digit.lexval

Question sources

Computer Science model exam · 2015 · Question 82

Question 17 | Q00631

Which one of the following can be applied directly after intermediate code generation to improve the intermediate representation?

A. Peephole optimization

B. Constant Propagation

C. Dead code elimination

D. All

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer D: All

Optimization changes intermediate code to remove inefficiencies (PDF p. 7). The module lists peephole transformations and constant propagation (p. 18), and elimination of never-executed or unreachable statements (p. 18). Each can improve an intermediate representation before target code is emitted; no exclusive ordering among those transformations is specified.

Why each choice is right or wrong

A - Incorrect: Peephole transformation improves instruction sequences (p. 18).

B - Incorrect: Constant propagation is a listed optimization technique (p. 18).

C - Incorrect: Removing unreachable intermediate statements is dead code elimination (p. 18).

D - Correct: All three fit the optimization of intermediate code described on p. 7.

Primary note proof

Study-note section COMP30

Constant folding evaluates constant expressions during compilation: 3*4→12. Constant propagation substitutes known values: x=3; y=x+2 becomes y=3+2 before folding. Copy propagation substitutes a copied name while its source remains unchanged. Dead-code elimination removes unreachable code or computations whose values are unused and whose evaluation has no necessary side effects.

Study-note section COMP31

Common-subexpression elimination reuses an earlier value only if operand values and relevant memory remain unchanged. A DAG can represent sharing. Algebraic simplification removes identities when valid: integer x+0→x. Strength reduction replaces an operation with an equivalent cheaper form, such as unsigned x*8→x<<3 for a valid width and shift count. Target costs determine whether it is actually cheaper.

Study-note section COMP31

Peephole optimization examines short instruction windows. It can remove redundant loads/stores, replace machine idioms and shorten jump chains: jump L1 where L1 immediately jumps L2 can become jump L2. Local optimization is a wider category; peephole optimization is a concrete method within the relevant scope.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 2-6

5. Code Optimization
Changes the IC by removing such inefficiencies
Improve the code
a. Improvement may be time, space, or power consumption.
It changes the structure of programs,

Compiler Design module | PDF page 18 | lines 23-34

Peephole Optimization
• Transforming to optimal sequence of instructions
Common Techniques:
• Elimination of redundant loads and stores
- Eg.
- r2 = r1 + 5
- I = r2
- r3 = I
- r4 = r3 * 3

• Constant folding
- Eg.
- R2 = 3 * 2

Compiler Design module | PDF page 18 | lines 32-41

• Constant folding
- Eg.
- R2 = 3 * 2

• Constant Propagation
- Eg.
- r1 = 3
- r2 = r1 * 2

• Copy Propagation
- Eg.
- r2 = r1

Compiler Design module | PDF page 18 | lines 2-10

Techniques
• Common sub-expression elimination
- Repeated appearance computed previously
• Strength reduction
- Replacement of expensive expressions with simple ones
• Code movement
- Moving a block of code outside a loop
• Dead code elimination
- Eliminated code statements that are either never executed or unreachable

Question sources

Computer Science model exam · 2015 · Question 100

Question 18 | Q00652

Which compiler structure associates declared identifier names or identities with attributes such as type, scope and storage location, irrespective of its container implementation?

A. Literal Table

B. Symbol table

C. Hash table

D. All

Hint

Ask whether the required information is a declaration’s identity and attributes, program structure or the ordered source tokens.

Answer B: Symbol table

A symbol table maps identifier names or declaration identities to attributes such as type, scope and storage location. A hash table is one possible implementation of that mapping, not its semantic role. Literal tables concern literal values, and punctuation need not have symbol records at all. Restricting the question to identifier metadata avoids claiming that every token kind uses the same table.

Why each choice is right or wrong

A - Incorrect: A literal table concerns literal values.

B - Correct: The symbol table provides this mapping.

C - Incorrect: Hashing is an implementation technique rather than the semantic role.

D - Incorrect: The structures have different roles.

Primary note proof

Study-note section COMP19

A symbol table maps names or symbol identities to attributes: type, scope, width, storage offset, parameter signature, constant information or linkage. A hash table gives expected O(1) insertion/lookup with suitable hashing; a stack of scope tables supports lexical nesting. Search the nearest scope first, insert declarations in the current scope, and remove visibility when leaving it.

Study-note section COMP19

Scanning may intern a spelling first; semantic analysis determines the declaration and type. Implementations differ, so first interaction is conventional rather than universal. Not every punctuation token needs a symbol-table entry; literals may use a separate literal table. A hash table is an implementation, not the information's semantic role.

Study-note section COMP41

Grouping characters into token classes is lexical analysis; grouping tokens into expressions and statements is syntax analysis.

Study-note section COMP41

Trees store hierarchical constructs, while a symbol table stores declaration metadata.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float

id.width=4
offset = offset +id.width { offset = 6 }

Compiler Design module | PDF page 8 | lines 10-18

Tokens, Patterns, and Lexemes
• Token
- A string of characters which logically belong together
- Classes of similar lexemes

l identifier, keywords, constants etc.
• Pattern
- A rule which describes a token

• Lexeme
- The sequence of characters matched by a pattern to form the token

Question sources

Computer Science model exam · 2015 · Question 121

Question 19 | Q00754

The output of lexical analyzer is*

A. Syntax tree

B. Strings of characters

C. A set of tokens

D. set of regular expressions

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: A set of tokens

The output of lexical analysis is expressly a sequence of tokens (PDF p. 8). C is the only option naming that output, although "set" should be read informally: order matters, so the module's precise word is sequence.

Why each choice is right or wrong

A - Incorrect: The parse tree is produced by syntax analysis.

B - Incorrect: Characters are the scanner input, not its classified output.

C - Correct: Tokens are the output elements (p. 8).

D - Incorrect: Patterns describe tokens; they are not the source-specific output sequence (p. 8).

Primary note proof

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Question sources

Computer Science model exam 2 · Question 95

Question 20 | Q00755

Which of the following is NOT the primary role of code generator in compiler design? *

A. instruction selection

B. Declaration of variables

C. instruction ordering

D. A Register allocation

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer B: Declaration of variables

Code generation lowers checked intermediate code into target operations. It selects instructions, allocates registers and orders instructions subject to dependencies. Declarations instead establish source names, types and storage information during analysis; generating code may use that information without declaring the variables again. Thus the declaration activity is outside the primary code-generator task set, while the other three choices describe target implementation decisions.

Why each choice is right or wrong

A - Incorrect: Instruction selection chooses target operations implementing the IR.

B - Correct: Processing source declarations belongs to analysis and symbol-table/type handling.

C - Incorrect: Instruction ordering schedules target computations while preserving dependencies.

D - Incorrect: Register allocation decides where intermediate values reside.

Primary note proof

Study-note section COMP26

> IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

> Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Study-note section COMP34

> Code generation lowers checked IR to a target form. Instruction selection chooses target operations; register allocation chooses where values reside; scheduling orders instructions while respecting dependencies; storage layout assigns offsets and follows ABI requirements. Register assignment names actual physical registers after allocation decisions.

Study-note section COMP34

> For t=a+b, a simple target could emit LOAD R1,a; LOAD R2,b; ADD R3,R1,R2; STORE t,R3. Other targets may combine loads with arithmetic or keep t in a register. Machine idioms, register availability and instruction order affect execution time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float
id.width=4
offset = offset +id.width { offset = 6 }

Question sources

Computer Science model exam 2 · Question 96

Question 21 | Q00756

Which one of the following is different from the others?

A. quadruples

B. triples

C. tree

D. indirect triples

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer C: tree

Quadruples, triples, indirect triples, trees and DAGs all implement three-address code (PDF p. 16). Quadruples and triples have numbered record fields, and indirect triples use pointers; tree is the hierarchical representation. C is the natural odd one out under that structural distinction.

Why each choice is right or wrong

A - Incorrect: Quadruples are instruction records with four fields (p. 16).

B - Incorrect: Triples are instruction records with three fields (p. 16).

C - Correct: Tree is the hierarchical structure among the options (p. 16).

D - Incorrect: Indirect triples retain a pointer-based triple representation (p. 16).

Primary note proof

Study-note section COMP26

> IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

> Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 16 | lines 2-8

> Three Address Code
> • Is a generic form and can be implemented as:
> - Quadruples
> - Triples
> - Indirect Triples
> - Tree
> - DAG

Compiler Design module | PDF page 16 | lines 13-23

> Three Address Code
> • Quadruples:
> - Each instruction is divided into four fields
> - Operator, arg1, arg2, and result
> • Triples:
> - Has three fields
> - Operator, arg1 and arg2
> • DAG and Tree
> - Similar presentation of expression to triples
> • Indirect Triples
> - Uses pointers instead of position to store results

Question sources

Computer Science model exam 2 · Question 97

Question 22 | Q00757

A compiler computes expression-type attributes and checks compatibility as grammar productions are processed. Which offered description most specifically identifies this mechanism?

A. During lexical analysis

B. During syntax analysis

C. During syntax directed translation

D. During code optimization

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: During syntax directed translation

The compiler in the scenario evaluates type attributes and applies compatibility checks as grammar productions are processed. This is syntax-directed translation: semantic rules or actions are attached to grammatical structure. Pure lexical analysis only recognizes tokens, and pure syntax recognition alone does not establish operand-type compatibility. Optimization transforms an already analyzed representation. Parsing can invoke these semantic actions, so the mechanism is more precise than naming syntax analysis alone.

Why each choice is right or wrong

A - Incorrect: Token recognition alone does not compute full expression-type compatibility.

B - Incorrect: Pure syntax recognition checks grammar; the scenario additionally attaches semantic computations.

C - Correct: Syntax-directed semantic actions compute and check attributes associated with productions.

D - Incorrect: Optimization transforms analyzed code rather than naming the stated checking mechanism.

Primary note proof

Study-note section COMP24

> A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions.

Study-note section COMP25

> A syntax-directed translation scheme embeds executable actions in productions; the actions run while translating a source program, not while generating the compiler. Attributes may compute types, values, AST pointers or generated code.

Study-note section COMP22

A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 12 | lines 4-16

Syntax Directed Translation
• Attaching actions to the grammar rules(productions).
• Actions are executed during the compilation
- Not during the generation of the compiler
• Actions are executed according to the parsing mechanism.
Syntax Directed Definitions
• Is a generalization of a context free grammar
• Is a CFG with attributes and rules
• Attributes are associated with grammar symbols and rules with productions
• Attributes may be:
- Numbers
- Types
- Strings etc

Question sources

Computer Science model exam 2 · Question 98

Question 23 | Q00758

In a conventional pipeline that lowers TAC further to native code or VM instructions, which representation is intermediate rather than the final emitted target?

A. Three address code

B. Assembly language

C. Absolute machine code

D. Byte Code

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer A: Three address code

Three-address code is intermediate in the specified pipeline: analysis and optimization transform it before code generation lowers it further. Assembly, absolute machine code and VM bytecode can each be final target forms for an appropriate compiler. This pipeline-specific distinction does not assert that TAC could never be another translator's target; it identifies the expressly stated intermediate role here.

Why each choice is right or wrong

A - Correct: TAC is lowered further in this pipeline.

B - Incorrect: Assembly can be a final emitted target.

C - Incorrect: Absolute machine code can be a final target.

D - Incorrect: VM bytecode can be a final target.

Primary note proof

Study-note section COMP34

> Code generation lowers checked IR to a target form. Instruction selection chooses target operations; register allocation chooses where values reside; scheduling orders instructions while respecting dependencies; storage layout assigns offsets and follows ABI requirements. Register assignment names actual physical registers after allocation decisions.

Study-note section COMP34

> For t=a+b, a simple target could emit LOAD R1,a; LOAD R2,b; ADD R3,R1,R2; STORE t,R3. Other targets may combine loads with arithmetic or keep t in a register. Machine idioms, register availability and instruction order affect execution time.

Study-note section COMP26

> IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

> Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 15 | lines 13-21

Three Address Code
• Is a sequence of statements of the form
- X = Y op Z
- X,Y and Z are names, constants or compiler generated temporaries
- Op is operator (arithmetic, logical)

• Example:
- a = b + c , x = -y , if a > b goto L1

• LHS is the target
• RHS has at most two sources and one operator

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Question sources

Computer Science model exam 2 · Question 99

Question 24 | Q00759

Which of the following is NOT important property of intermediate code representation?

A. Ease of generation

B. Ease of manipulation

C. Register allocation

D. Level of abstraction

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer C: Register allocation

An intermediate representation should be convenient to generate, analyze and transform, and its abstraction should suit optimization and target lowering. Those are design properties of the representation. Register allocation is an algorithmic backend task that chooses locations for values; it is not itself a property such as ease of generation or abstraction level. An IR can carry information useful to that task without making allocation a representation property.

Why each choice is right or wrong

A - Incorrect: Ease of generation is a useful IR design property.

B - Incorrect: Ease of manipulation supports analysis and transformation passes.

C - Correct: Register allocation is a backend decision over values rather than an IR property.

D - Incorrect: Abstraction level determines which source and target details the representation exposes.

Primary note proof

Study-note section COMP26

> IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

> Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Study-note section COMP34

> Code generation lowers checked IR to a target form. Instruction selection chooses target operations; register allocation chooses where values reside; scheduling orders instructions while respecting dependencies; storage layout assigns offsets and follows ABI requirements. Register assignment names actual physical registers after allocation decisions.

Study-note section COMP34

> For t=a+b, a simple target could emit LOAD R1,a; LOAD R2,b; ADD R3,R1,R2; STORE t,R3. Other targets may combine loads with arithmetic or keep t in a register. Machine idioms, register availability and instruction order affect execution time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 11-12

> 4. Intermediate Code Generation
> Easy to produce and easy to translate to machine code

Compiler Design module | PDF page 7 | lines 7-11

> 6. Code Generation
> Converts intermediate code to machine code.
> Must handle all aspects of machine architecture
> Storage allocation decisions are made
> a. Register allocation and assignment

Compiler Design module | PDF page 16 | lines 2-8

> Three Address Code
> • Is a generic form and can be implemented as:
> • Quadruples

- Triples
- Indirect Triples
- Tree
- DAG

Question sources

Computer Science model exam 2 · Question 100

Question 25 | Q00903

What is the main purpose of syntax-directed translation in a compiler?

A. To interleave semantic analysis with syntax analysis using attributes attached to grammar symbols

B. To generate optimized machine code directly

C. .
To tokenize the source code

D. To replace the parser with a finite automaton

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer A: To interleave semantic analysis with syntax analysis using attributes attached to grammar symbols

A syntax-directed definition is a CFG with attributes and rules; attributes attach to symbols and rules to productions (PDF p. 12). Actions execute according to the parsing mechanism. This is the grammar-based integration of semantic processing described in A, rather than a direct optimized-machine-code replacement for the remaining phases.

Why each choice is right or wrong

A - Correct: A matches the attribute/rule and parsing-action definitions (p. 12).

B - Incorrect: Machine-code conversion is a later code-generation task (p. 7), not the definition of SDT.

C - Incorrect: Tokenization is lexical analysis (p. 8).

D - Incorrect: The module retains the grammar/parsing mechanism; it does not replace it with an automaton (p. 12).

Primary note proof

Study-note section COMP24

A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions. An annotated parse tree displays computed attribute values; it is not itself an

inherited or synthesized attribute. Synthesized attributes flow from children to a parent; inherited attributes carry context from parent or siblings.

Study-note section COMP24

For E→E1+T, E.val=E1.val+T.val synthesizes an expression value. A declaration's type environment passed into its body is inherited. S-attributed definitions use only synthesized attributes and fit LR reduction-time evaluation naturally; suitable LL parsers can also compute them. L-attributed definitions restrict dependencies so a left-to-right traversal can evaluate inherited context. Attribute dependency cycles can prevent evaluation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 4-16

Syntax Directed Translation
• Attaching actions to the grammar rules(productions).
• Actions are executed during the compilation
- Not during the generation of the compiler
• Actions are executed according to the parsing mechanism.
Syntax Directed Definitions
• Is a generalization of a context free grammar
• Is a CFG with attributes and rules
• Attributes are associated with grammar symbols and rules with productions
• Attributes may be:
- Numbers
- Types
- Strings etc

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Computer Science Exit Exam · 2018 · Question 16

Question 26 | Q00908

Straight directed translation uses:

A. Purely lexical rules

B. Backtracking algorithms

C. ¢.
Grammar without attributes

D. Grammar with sematic rule

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer D: Grammar with sematic rule

Interpreting the apparent typo “Straight directed translation” as syntax-directed translation, the module defines it through actions on productions and a CFG with attributes and semantic rules (PDF p. 12). D is the only option that retains this grammar-plus-rule structure.

Why each choice is right or wrong

A - Incorrect: Pure lexical rules omit the production-associated semantic actions (p. 12).

B - Incorrect: Backtracking is not the defining mechanism; actions follow the chosen parser (p. 12).

C - Incorrect: A grammar without attributes omits the defining additions in a syntax-directed definition (p. 12).

D - Correct: A grammar with semantic rules matches the definition (p. 12).

Primary note proof

Study-note section COMP24

> A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions. An annotated parse tree displays computed attribute values; it is not itself an inherited or synthesized attribute. Synthesized attributes flow from children to a parent; inherited attributes carry context from parent or siblings.

Study-note section COMP24

> For E→E1+T, E.val=E1.val+T.val synthesizes an expression value. A declaration’s type environment passed into its body is inherited. S-attributed definitions use only synthesized attributes and fit LR reduction-time evaluation naturally; suitable LL parsers can also compute them. L-attributed definitions restrict dependencies so a left-to-right traversal can evaluate inherited context. Attribute dependency cycles can prevent evaluation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 4-16

Syntax Directed Translation
• Attaching actions to the grammar rules(productions).
• Actions are executed during the compilation
- Not during the generation of the compiler
• Actions are executed according to the parsing mechanism.
Syntax Directed Definitions
• Is a generalization of a context free grammar
• Is a CFG with attributes and rules
• Attributes are associated with grammar symbols and rules with productions
• Attributes may be:
- Numbers
- Types
- Strings etc

Question sources

Computer Science Exit Exam · 2018 · Question 21

Question 27 | Q00911

Which one of the following is a top-down parser without backtracking?

A. LR parser

B. Brute force parser

C. Operator precedence parser

D. Predictive parser

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer D: A predictive parser.

COMP11 defines predictive parsing by choosing productions with lookahead without backtracking. This supplies both requirements in the stem: top-down direction and absence of backtracking. LR and operator-precedence parsing belong to bottom-up approaches; brute-force top-down trial does not provide the no-backtracking guarantee.

Why each choice is right or wrong

A - Incorrect: LR parsing reduces input bottom-up rather than expanding the start symbol top-down.

B - Incorrect: Brute-force production trials can require backtracking and do not meet the defining predictive constraint.

C - Incorrect: Operator-precedence parsing is a bottom-up technique rather than predictive top-down expansion.

D - Correct: Predictive parsing uses lookahead to choose productions without backtracking, exactly as COMP11 states.

Primary note proof

Study-note section COMP11

> Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP15

> Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse.

Study-note section COMP-S01

> Operator-precedence parsing is a restricted bottom-up expression-parsing method using a stack and precedence relations. With a suitable expression grammar, the stack retains unfinished surrounding expressions while nested parentheses are processed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

> Parsing
> • Process of constructing parse tree for a sentence generated by a given grammar.
> • 2 types of parsers
> ● Top down parsing (predictive parsers)
> • LL(1)
> ● Bottom up parsing (SR parsers)
> • LR(1)

Compiler Design module | PDF page 10 | lines 18-23

> Top Down Parsing
> • The parse tree is created top to bottom
> • Starts from the start symbol and transform it to the input
> Bottom Up Parsing
> • Starts with the input symbols and tries to construct the parse tree up to the start symbol.
> • One way of reducing a sentence is to follow the right most derivation in reverse

Compiler Design module | PDF page 11 | lines 2-17

> LL(1) Grammar
> • L - left to right
> • L - left most derivation
> • 1 - number of look ahead
> • First( ) and Follow( )

- the first terminal in a string and the terminal that follows a variable respectively.

LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)

LL	LR
Leftmost derivation	Rightmost derivation in reverse
Starts with root non-terminal on stack	Ends with root non-terminal on the stack
Builds the parse tree top-down	Builds the parse tree bottom-up
Expands the non-terminals	Reduces the non-terminals

Question sources

Computer Science Exit Exam · 2018 · Question 24

Question 28 | Q00940

If input is "a" is for syntax tree, which function is used to create tree?

A. mknode(num, value)

B. mkleaf(num, value)

C. mkleaf(id, entry)

D. mknode(op, left, right)

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer C: mkleaf(id, entry)

For the expression a-4+c the first construction is P1=mkleaf(id,entrya) (PDF p. 13). Since a is an identifier, its node stores a pointer to the identifier’s symbol-table entry. Numeric leaf construction is instead used for 4; operator nodes connect child nodes.

Why each choice is right or wrong

A - Incorrect: The module uses mkleaf, not mknode, for a numeric constant, and a is not a number (p. 13).

B - Incorrect: This creates a numeric leaf, suitable for 4 rather than identifier a (p. 13).

C - Correct: This is exactly the identifier-leaf constructor used for a (p. 13).

D - Incorrect: This makes an operator node with two children, not an identifier leaf (pp. 12-13).

Primary note proof

Study-note section COMP25

A syntax-directed translation scheme embeds executable actions in productions; the actions run while translating a source program, not while generating the compiler. Attributes may compute types, values, AST pointers or generated code. The scheme defines an input-to-output translation, not merely an input language.

Study-note section COMP25

Create an identifier leaf with mkleaf(id,entry), a number with mkleaf(num,value), and an operator node with mknode(op,left,right). For a−4+c, construct +(−(a,4),c). Postorder traversal prints postfix: a 4 − c +. A stack evaluator pushes operands and pops right then left operands for an operator. Infix-to-postfix is useful for stack machines but not mandatory for all compilers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 26-28

Functions for Syntax Tree Nodes
• mknode (op, left, right)
- Creates an operator node with label op &

Compiler Design module | PDF page 13 | lines 2-14

- Two fields containing pointers to left and right
• mkleaf(id, entry)
- Creates an identifier node with label id &
- A field containing entry, ptr to symbol table entry for the identifier
• mkleaf(num, val)
- Create a number node with label num &
- A field containing val, the value for the number

Syntax tree for expression a-4+c
• P1=mkleaf(id,entrya);
• P2=mkleaf(num, 4);
• P3=mknode('-',p1,p2);
• P4=mkleaf(id,entryc);
• P5=mknode('+',p3,p4);

Question sources

Computer Science Exit Exam · 2018 · Question 53

Question 29 | Q00945

Which statement is WRONGLY described about code generation?

A. Code generation can be considered as the start phase of compilation

B. The target program is the output of the code generator

C. Checked intermediate code is supplied as input to the code generator.

D. It used to produce the target code for three-address statements.

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Code generation can be considered as the start phase of compilation

Lexical analysis begins the standard pipeline by recognizing tokens. Code generation occurs after analysis and IR construction, lowering checked IR to a target form. Its output is target code, which may be assembly or an object form, and it can translate three-address statements. Therefore calling code generation the start phase is the wrong description. The input option is read as checked IR suitable for lowering, without claiming a compiler can prove the absence of every program defect.

Why each choice is right or wrong

A - Correct: Code generation follows the frontend and IR; it is not the initial scanning phase.

B - Incorrect: The generator produces code in a target representation.

C - Incorrect: Checked intermediate code supplies the operations and metadata to lower.

D - Incorrect: Three-address statements can be lowered into target instructions.

Primary note proof

Study-note section COMP03

> The conventional sequence is lexical analysis → syntax analysis → semantic analysis → intermediate-code generation → optimization → target-code generation. Scanning turns characters into tokens; parsing builds structure; semantics checks declarations and types; synthesis produces implementation code. A phase is a logical activity; a pass is a traversal that can combine activities.

Study-note section COMP34

> Code generation lowers checked IR to a target form.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 7-11

> 6. Code Generation
> Converts intermediate code to machine code.
> Must handle all aspects of machine architecture
> Storage allocation decisions are made
> a. Register allocation and assignment

Compiler Design module | PDF page 5 | lines 3-11

> 1. Lexical / Linear Analysis (scanning)
> - Scans the source code as a stream of characters
> - Represent lexemes in the form of tokens as:
> <token-name, attribute-value>

- Token
- smallest meaningful element that a compiler understands.
- Eg.
- Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Question sources

Computer Science Exit Exam · 2018 · Question 58

Question 30 | Q00973

Which of the following is NOT a task of the lexical analyzer?

A. Generating parse trees

B. Correlating errors with line numbers

C. Entering identifiers into the symbol table

D. Stripping whitespace and comments

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Generating parse trees

Scanning turns characters into tokens, usually skips insignificant whitespace and comments, and retains source positions for diagnostics. A scanner may also intern identifier spellings in a symbol table; later semantic processing adds scope and declaration information. Parse-tree construction belongs to syntax analysis because it arranges the token stream according to grammar productions. It is consequently the activity outside the scanner's listed responsibilities.

Why each choice is right or wrong

A - Correct: Parse trees represent grammar structure built by parsing.

B - Incorrect: Tracking token positions and line numbers supports scanner diagnostics.

C - Incorrect: Scanner-side interning can enter identifier spellings, distinct from later declaration analysis.

D - Incorrect: Ordinary scanners discard insignificant whitespace and comments.

Primary note proof

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float
id.width=4
offset = offset +id.width { offset = 6 }

Question sources

Computer Science Exit Exam · 2018 · Question 86

Question 31 | Q01165

Which analysis used to read the sequence of the characters from the source program?

A. a | Lexical Analysis A

B. i? Syntax Analysis ><

C. /@| Semantic Analysis x

D. | All of the above x

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: a | Lexical Analysis A

PDF p. 5 explicitly says lexical analysis scans source code as a stream of characters; syntax groups tokens, and semantics checks meaningful compatibility (p. 6).

Why each choice is right or wrong

A - Correct: Lexical analysis explicitly reads the character stream.

B - Incorrect: Syntax analysis works on tokens grouped hierarchically.

C - Incorrect: Semantic analysis performs meaning/type checks (p. 6).

D - Incorrect: The raw-character scanning responsibility is assigned specifically to lexical analysis, not all three phases (pp. 5-6).

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

> 1. Lexical / Linear Analysis (scanning)
> - Scans the source code as a stream of characters

- Represent lexemes in the form of tokens as:
<token-name, attribute-value>
- Token
• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Computer Science Exit Exam practice questions · Question 26

Question 32 | Q01193

What is the bottom-up parsing method also known as?

A. Recursive descent parsing

B. Predictive parsing

C. Shift-reduce parsing

D. All of them

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer C: Shift-reduce parsing

Bottom-up parsing starts with input symbols and reduces recognized right-hand sides until it reaches the start symbol. A shift-reduce parser implements this by shifting input onto a stack and reducing handles. Recursive descent expands from a start-symbol procedure and is top-down; predictive parsing is a top-down approach that uses lookahead to choose productions. Therefore the aggregate alternative cannot apply to methods with these different construction directions.

Why each choice is right or wrong

A - Incorrect: Recursive descent expands productions from nonterminals and is top-down.

B - Incorrect: Predictive parsing chooses top-down productions using lookahead.

C - Correct: Shifting input and reducing handles is the bottom-up shift-reduce method.

D - Incorrect: The list includes top-down methods, so not all are bottom-up.

Primary note proof

Study-note section COMP11

> Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP15

> Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

> Parsing
> • Process of constructing parse tree for a sentence generated by a given grammar.
> • 2 types of parsers
> ● Top down parsing (predictive parsers)
> • LL(1)
> ● Bottom up parsing (SR parsers)
> • LR(1)

Compiler Design module | PDF page 10 | lines 18-23

> Top Down Parsing
> • The parse tree is created top to bottom
> • Starts from the start symbol and transform it to the input
> Bottom Up Parsing
> • Starts with the input symbols and tries to construct the parse tree up to the start symbol.
> • One way of reducing a sentence is to follow the right most derivation in reverse

Question sources

Computer Science model exam 2 · Question 14

Question 33 | Q01222

How many tokens are in the C++ loop header for (int i = 0; i <= 10; i++)? Ignore whitespace and count <= and ++ as single operator tokens.

A. 4

B. 5

C. 9

D. 14

Hint

Follow the representation at each construction step and apply the pattern or state-set rules consistently, including empty cases.

Answer D: 14

The tokens are for, (, int, i, =, 0, ;, i, <=, 10, ;, i, ++,). There are fourteen occurrences. Whitespace contributes nothing, but both semicolons and both parentheses count. Longest match makes <= and ++ one token each; it does not merge neighboring identifiers or discard repeated tokens. The numerical option therefore gives the unique count.

Why each choice is right or wrong

A - Incorrect: Four omits most token occurrences.

B - Incorrect: Five misses operators, punctuation and operands.

C - Incorrect: Nine is below the fourteen-token count.

D - Correct: The complete sequence contains fourteen tokens.

Primary note proof

Study-note section COMP07

> At each input position, choose the longest matching lexeme; if equal lengths match multiple flex rules, choose the earliest rule. Thus ifx is an identifier rather than if followed by x; <= and ++ are each one operator token.

Study-note section COMP07

> There are 14 tokens, irrespective of spacing. The increment operator is one token.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 10-18

Tokens, Patterns, and Lexemes
• Token
- A string of characters which logically belong together
- Classes of similar lexemes

l identifier, keywords, constants etc.
• Pattern
- A rule which describes a token

• Lexeme
- The sequence of characters matched by a pattern to form the token

Compiler Design module | PDF page 8 | lines 19-24

• Classes of Tokens
- Identifiers: names chosen by the programmer
- Keywords: names already in the programming language
- Separators: punctuation characters
- Operators: symbols that operate on arguments and produce results
- Literals: numeric, textual literals

Question sources

Computer Science model exam 2 · Question 43

Question 34 | Q01243

Identify the correct definition of lexical analysis.

A. Breaking a sequence of characters into packets.

B. Breaking a sequence of characters into groups.

C. Breaking a sequence of characters into tokens.

D. Breaking a sequence of characters into lines.

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: Breaking a sequence of characters into tokens.

Lexical analysis takes a source program and outputs a sequence of tokens (PDF p. 8). Thus its relevant grouping units are tokens, rather than network packets, arbitrary groups, or source lines.

Why each choice is right or wrong

A - Incorrect: Packets are not the compiler units described on p. 8.

B - Incorrect: Groups is too unspecific; the defined output units are tokens (p. 8).

C - Correct: Breaking the character sequence into tokens matches scanning (pp. 5, 8).

D - Incorrect: Lines are tracked and whitespace stripped; they are not the primary output units (p. 8).

Primary note proof

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Compiler Design module | PDF page 8 | lines 10-18

Tokens, Patterns, and Lexemes
• Token
- A string of characters which logically belong together
- Classes of similar lexemes
l identifier, keywords, constants etc.
• Pattern
- A rule which describes a token
• Lexeme
- The sequence of characters matched by a pattern to form the token

Question sources

Computer Science model exam 2 · Question 64

Question 35 | Q01254

What does a top-down parser generate?

A. Rightmost derivation in reverse.

B. Rightmost derivation.

C. Leftmost derivation in reverse.

D. Leftmost derivation.

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer D: Leftmost derivation.

The module says top-down parsing builds from the start symbol (PDF p. 10), and the LL/LR comparison pairs top-down LL with leftmost derivation and bottom-up LR with rightmost derivation in reverse (p. 11). D matches the top-down direction.

Why each choice is right or wrong

A - Incorrect: Rightmost derivation in reverse is the LR/bottom-up entry (p. 11).

B - Incorrect: Rightmost derivation alone is not the listed top-down entry (p. 11).

C - Incorrect: The top-down derivation is leftmost in the forward direction, not reversed (p. 11).

D - Correct: Leftmost derivation is explicitly the LL/top-down entry (p. 11).

Primary note proof

Study-note section COMP11

> Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP11

> A nonrecursive predictive parser stores grammar symbols on an explicit stack and consults a table. It is another implementation of predictive parsing, not a wholly different language class. Table-driven and recursive predictive implementations can both be linear for a fixed suitable grammar. Efficiency cannot be ranked from their names alone.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 18-23

> Top Down Parsing
> • The parse tree is created top to bottom

• Starts from the start symbol and transform it to the input
Bottom Up Parsing
• Starts with the input symbols and tries to construct the parse tree up to the start symbol.
• One way of reducing a sentence is to follow the right most derivation in reverse

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science model exam 2 · Question 75

Question 36 | Q01257

For line 3, fro ((I = o, I;, in the displayed C program, classify only lexical and syntactic well-formedness. Treat fro and o as legal identifier spellings and leave declaration/type diagnostics outside this question.

A. Syntax error, with no lexical error established.

B. No compilation error.

C. Only lexical error.

D. Both lexical and syntax error.

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Syntax error, with no lexical error established.

The characters fro and o form valid identifier lexemes, so their spellings do not establish a lexical error. The displayed line nevertheless has mismatched parentheses and an invalid token arrangement, establishing a syntax error. Under this restricted lexical/syntactic classification, A is correct. Declaration or type diagnostics may arise in a real compiler, but are outside the expressly stated classification scope.

Why each choice is right or wrong

A - Correct: The arrangement is syntactically invalid without an established lexical fault.

B - Incorrect: Mismatched delimiters prevent syntactic validity.

C - Incorrect: Legal identifier spellings do not establish a lexical error.

D - Incorrect: A syntax error exists but no lexical error is established.

Primary note proof

Study-note section COMP08

> A lexical error prevents forming a legal token, such as an invalid character or unterminated literal. A syntax error is an illegal token arrangement, such as a missing delimiter. A semantic error violates name, scope or type rules after grammatical recognition.

Study-note section COMP08

> In the malformed C line fro ((I = o, I;, fro and lowercase o are legal identifier lexemes. Misspelling for does not turn fro into a lexical error; mismatched syntax is definite, and undeclared o can also produce a semantic diagnostic.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

> 1. Lexical / Linear Analysis (scanning)
> - Scans the source code as a stream of characters
> - Represent lexemes in the form of tokens as:
> <token-name, attribute-value>
> - Token
> • smallest meaningful element that a compiler understands.
> - Eg.
> • Identifiers, Keywords, Literals, Operators and Special symbols.
> - Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 5 | lines 12-18

> 2. Syntax / Hierarchical Analysis - Parsing
> • Tokens are grouped hierarchically into nested collections with collective meaning.
> • The result is generally a parse tree.
> • expressions, statements, declarations etc... are identified by using the results of lexical analysis.

• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 8 | lines 10-18

Tokens, Patterns, and Lexemes
• Token
● A string of characters which logically belong together
● Classes of similar lexemes
l identifier, keywords, constants etc.
• Pattern
● A rule which describes a token
• Lexeme
● The sequence of characters matched by a pattern to form the token

Question sources

Computer Science model exam 2 · Question 78

Question 37 | Q01397

Which tool is used for grouping of characters in tokens in the compiler?

A. Parser

B. Code optimizer

C. Code generator

D. Scanner

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Scanner

Lexical analysis is called scanning on PDF p. 5 and outputs tokens on p. 8. Hence the scanner is the component that groups source characters into tokens, before the parser handles token structure.

Why each choice is right or wrong

A - Incorrect: The parser groups tokens into syntactic constructs.

B - Incorrect: The optimizer improves intermediate code (p. 7).

C - Incorrect: The generator converts intermediate code to machine code (p. 7).

D - Correct: Scanner is the component associated with lexical analysis (pp. 5, 8).

Primary note proof

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

1. Lexical / Linear Analysis (scanning)
- Scans the source code as a stream of characters
- Represent lexemes in the form of tokens as:
<token-name, attribute-value>
- Token
• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Question sources

Compiler Design review questions · Question 5

Question 38 | Q01399

When parsing techniques are classified by the direction in which they construct the parse tree, how many broad categories are there?

A. three types

B. four types

C. two types

D. five types

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer C: two types

Construction direction gives two broad categories. Top-down parsing expands from the start symbol toward the input, while bottom-up parsing reduces input toward the start symbol. Recursive descent, predictive parsing and LR are algorithms or families within that classification rather than additional directions. Counting named algorithms would give a different and open-ended taxonomy, so the revised question explicitly counts only the broad construction directions.

Why each choice is right or wrong

A - Incorrect: Three is not the number of construction directions in this two-way taxonomy.

B - Incorrect: Four may count selected algorithms, but it does not count the two broad directions.

C - Correct: The directions are top-down and bottom-up.

D - Incorrect: Five is not the number of construction directions.

Primary note proof

Study-note section COMP-S01

> The common two-way classification of parsing is by construction direction: top-down parsing expands from the start symbol, while bottom-up parsing reduces the input toward that symbol. This counts broad directions, not the number of parser algorithms or implementations.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

> Parsing
> • Process of constructing parse tree for a sentence generated by a given grammar.
> • 2 types of parsers
> ● Top down parsing (predictive parsers)
> • LL(1)
> ● Bottom up parsing (SR parsers)
> • LR(1)

Question sources

Compiler Design review questions · Question 7

Question 39 | Q01400

In which parsing, the parser constructs the parse tree from the start symbol and transforms it into the input symbol.

A. Bottom-up parsing

B. Top-down parsing

C. None of the above

D. Both a and b

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: Top-down parsing

Top-down parsing starts from the start symbol and transforms it into the input (PDF p. 10). Bottom-up parsing starts with input symbols and constructs toward the start symbol. The stem is an almost verbatim description of the first direction.

Why each choice is right or wrong

A - Incorrect: Bottom-up starts from input symbols, the opposite endpoint (p. 10).

B - Correct: Top-down starts from the start symbol as requested (p. 10).

C - Incorrect: The described category is explicitly present, so none is false (p. 10).

D - Incorrect: Both is false because the two categories have opposite starting points (p. 10).

Primary note proof

Study-note section COMP11

> Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP11

> A nonrecursive predictive parser stores grammar symbols on an explicit stack and consults a table. It is another implementation of predictive parsing, not a wholly different language class. Table-driven and recursive predictive implementations can both be linear for a fixed suitable grammar. Efficiency cannot be ranked from their names alone.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 18-23

Top Down Parsing
• The parse tree is created top to bottom
• Starts from the start symbol and transform it to the input
Bottom Up Parsing
• Starts with the input symbols and tries to construct the parse tree up to the start symbol.
• One way of reducing a sentence is to follow the right most derivation in reverse

Question sources

Compiler Design review questions · Question 8

Question 40 | Q01401

Which derivation is generated by the top-down parser?

A. Right-most derivation in reverse

B. Left-most derivation in reverse

C. Right-most derivation

D. Left-most derivation

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer D: Left-most derivation

The LL/top-down column uses leftmost derivation, while the LR/bottom-up column uses rightmost derivation in reverse (PDF p. 11). Consequently a top-down parser generates a forward leftmost derivation.

Why each choice is right or wrong

A - Incorrect: This is the bottom-up LR direction (p. 11).

B - Incorrect: The LL entry is leftmost without reversal (p. 11).

C - Incorrect: Rightmost is not the listed top-down derivation (p. 11).

D - Correct: Leftmost is explicitly the top-down/LL entry (p. 11).

Primary note proof

Study-note section COMP11

Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP11

A nonrecursive predictive parser stores grammar symbols on an explicit stack and consults a table. It is another implementation of predictive parsing, not a wholly different language class. Table-driven and recursive predictive implementations can both be linear for a fixed suitable grammar. Efficiency cannot be ranked from their names alone.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Compiler Design module | PDF page 10 | lines 18-23

Top Down Parsing
• The parse tree is created top to bottom
• Starts from the start symbol and transform it to the input
Bottom Up Parsing
• Starts with the input symbols and tries to construct the parse tree up to the start symbol.
• One way of reducing a sentence is to follow the right most derivation in reverse

Question sources

Compiler Design review questions · Question 9

Question 41 | Q01402

Which derivation is generated by the bottom-up parser?

A. Right-most derivation in reverse

B. Left-most derivation in reverse

C. Right-most derivation

D. Left-most derivation

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer A: Right-most derivation in reverse

PDF p. 10 describes bottom-up parsing as reducing input symbols toward the start symbol, following a rightmost derivation in reverse. PDF p. 11 repeats this as the definition of LR parsing. Thus A gives both the correct derivation kind and direction.

Why each choice is right or wrong

A - Correct: Rightmost derivation in reverse is explicitly stated (pp. 10-11).

B - Incorrect: Leftmost in reverse is not the module's bottom-up rule (p. 11).

C - Incorrect: Rightmost without reversal would expand rather than describe the stated reductions (pp. 10-11).

D - Incorrect: Leftmost is paired with LL/top-down parsing (p. 11).

Primary note proof

Study-note section COMP15

> Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse. A handle is a production RHS occurrence whose reduction is the next step of that reverse derivation. Not every substring matching a RHS is a handle in the current context.

Study-note section COMP15

> Shift pushes the next input token and a state; reduce replaces a recognized RHS by its LHS and consults GOTO; accept recognizes completion; error reports an invalid situation. For A→XYZ, pop three grammar symbols and their states, expose state q, then push A and GOTO[q,A]. "Go" is not a separate standard shift-reduce ACTION, although GOTO is essential table machinery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 18-23

> Top Down Parsing
> • The parse tree is created top to bottom
> • Starts from the start symbol and transform it to the input
> Bottom Up Parsing
> • Starts with the input symbols and tries to construct the parse tree up to the start symbol.
> • One way of reducing a sentence is to follow the right most derivation in reverse

Compiler Design module | PDF page 11 | lines 2-17

> LL(1) Grammar
> • L - left to right
> • L - left most derivation
> • 1 - number of look ahead

• First( ) and Follow( )
• the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Compiler Design review questions · Question 10

Question 42 | Q01403

Among these named deterministic parser methods, which has the broadest grammar coverage under the standard SLR(1), LALR(1) and canonical LR(1) comparison?

A. Operator Precedence

B. SLR

C. Canonical LR

D. LALR

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer C: Canonical LR

Canonical LR(1) keeps context-specific lookahead and covers at least the grammars handled by SLR(1) and LALR(1), with strictly greater coverage in the standard comparison.
Operator-precedence parsing is a more restricted expression-parsing method. This is a comparison of grammar coverage, not speed, implementation simplicity or table size. Canonical LR may require larger tables and still does not parse every unambiguous context-free grammar.

Why each choice is right or wrong

A - Incorrect: Operator-precedence parsing applies to a more restricted grammar family.

B - Incorrect: SLR(1) uses less context-specific reduction information and has narrower grammar coverage.

C - Correct: Canonical LR(1) has the broadest coverage among the named methods.

D - Incorrect: Merging LR(1) cores for LALR(1) can lose distinctions needed by some LR(1) grammars.

Primary note proof

Study-note section COMP-S01

> In the standard comparison by grammar coverage, canonical LR(1) is more general than the restricted operator-precedence method. Together with SLR(1)⊆LALR(1)⊆LR(1), this makes canonical LR(1) the broadest of those named deterministic parsing methods. Broader coverage does not mean faster execution or smaller tables.

Study-note section COMP17

> The standard grammar-class hierarchy is LR(0)⊆SLR(1)⊆LALR(1)⊆LR(1), with increasing capability and often table size.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 11 | lines 2-17

> LL(1) Grammar
> • L - left to right
> • L - left most derivation
> • 1 - number of look ahead
> • First( ) and Follow( )
> ● the first terminal in a string and the terminal that follows a variable respectively.
> LR Parsing
> • LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
> Types of LR Parsers
> • LR (0) , SLR (1) , LALR (1) , CLR (1)
> LL LR
> Leftmost derivation Rightmost derivation in reverse
> Starts with root non-terminal on stack Ends with root non-terminal on the stack
> Builds the parse tree top-down Builds the parse tree bottom-up
> Expands the non-terminals Reduces the non-terminals

Question sources

Compiler Design review questions · Question 11

Question 43 | Q01404

The output of the lexical analyzer is _____

A. string character

B. a syntax tree

C. a set of RE

D. a set of tokens

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: a set of tokens

The lexical analyzer's output is a sequence of tokens (PDF p. 8). Among the options only D identifies these elements; "set" is informal wording for the ordered stream.

Why each choice is right or wrong

A - Incorrect: Source characters are scanned as input.

B - Incorrect: The syntax tree belongs to parsing.

C - Incorrect: Regular expressions are not the listed scanner output; patterns are rules (p. 8).

D - Correct: Tokens are explicitly the scanner output (p. 8).

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 4-9

> What is Lexical Analysis
> • The first phase of a compiler
> • The input is a high level language program
> • The output is a sequence of tokens
> • Strips off blanks, tabs, newlines, and comments from the source program
> • Keeps track of line numbers

Compiler Design module | PDF page 8 | lines 10-18

> Tokens, Patterns, and Lexemes
> • Token
> • A string of characters which logically belong together

- Classes of similar lexemes

l identifier, keywords, constants etc.

• Pattern

- A rule which describes a token

• Lexeme

- The sequence of characters matched by a pattern to form the token

Question sources

Compiler Design review questions · Question 12

Question 44 | Q01405

Which is the least sufficient Chomsky grammar family for ordinary token patterns specified by classical regular expressions, rather than a purpose-based grammar name?

A. Lexical Grammar

B. Context-free Grammar

C. Syntactic Grammar

D. Regular Grammar

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer D: Regular Grammar

Ordinary regular-expression token patterns describe regular languages, whose least sufficient Chomsky family is regular grammar. A lexical grammar names a specification by purpose, whereas this question asks for its formal family. Context-free grammars are more expressive than required and typically model nested syntax. Syntactic grammar is another purpose label rather than the specific formal family matching the regular token patterns.

Why each choice is right or wrong

A - Incorrect: Lexical grammar is a purpose label, not the requested family.

B - Incorrect: CFGs are stronger than these regular patterns require.

C - Incorrect: Syntactic grammar is a purpose label.

D - Correct: Regular grammars generate exactly regular token languages.

Primary note proof

Study-note section COMP-S03

Classical regular expressions, consistently right-linear regular grammars and finite automata describe the same regular-language family. Thus an ordinary token pattern specified by a regular expression also has a regular grammar and an equivalent finite-state recognizer.

Study-note section COMP-S03

This equivalence concerns expressive power, not spelling of the specification or implementation size. Context-free grammars can also describe regular languages, but their larger family is needed for syntax with unbounded nested structure rather than ordinary regular token patterns.

Study-note section COMP18

Regular patterns are chiefly lexical specifications; CFGs chiefly describe nested syntax.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 6-16

Grammars
• Every programming language has grammar rules
• Parsers or syntax analyzers are generated for a particular grammar
• CFG are used for syntax specification of programming languages
Context Free Grammar (CFG)
• Is denoted as G = (N, T , P, S)
• N : finite set of non-terminals
• T : finite set of terminals
• S ∈ N: The start symbol
• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Compiler Design module | PDF page 8 | lines 10-18

Tokens, Patterns, and Lexemes
• Token
- A string of characters which logically belong together
- Classes of similar lexemes
l identifier, keywords, constants etc.
• Pattern
- A rule which describes a token
• Lexeme
- The sequence of characters matched by a pattern to form the token

Question sources

Compiler Design review questions · Question 13

MOE Computer Science Exit Exam · 2016 · Question 93

Question 45 | Q01407

In Compiler lexical analyzer is used for?

A. removing comments

B. removing whitespace

C. breaking the syntaxes in the set of tokens

D. All of the mentioned

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: All of the mentioned

Lexical analysis scans source characters into tokens and removes blanks/newlines/comments (PDF pp. 5, 8). Thus comment removal, whitespace removal, and tokenization all belong to the scanner in this module, making D the inclusive choice.

Why each choice is right or wrong

A - Incorrect: Comment removal is expressly stated (p. 8).

B - Incorrect: Whitespace stripping is expressly stated (p. 8).

C - Incorrect: Interpreting "syntaxes" as source text, tokenization is the lexical role.

D - Correct: All combines the three documented responsibilities (pp. 5, 8).

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

> 1. Lexical / Linear Analysis (scanning)
> - Scans the source code as a stream of characters
> - Represent lexemes in the form of tokens as:

<token-name, attribute-value>
- Token
• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Question sources

Compiler Design review questions · Question 15

Question 46 | Q01408

Which is considered as the sequence of characters in a token?

A. Mexeme

B. Lexeme

C. Texeme

D. Pattern

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Lexeme

PDF p. 8 calls the matched character sequence a Lexeme. A pattern is the rule that matches that sequence, rather than the concrete spelling itself.

Why each choice is right or wrong

A - Incorrect: Mexeme is not a defined tokenization term (p. 8).

B - Correct: Lexeme is the exact defined term (p. 8).

C - Incorrect: Texeme is not a defined tokenization term (p. 8).

D - Incorrect: Pattern is a rule describing a token, not the matched sequence (p. 8).

Primary note proof

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 10-18

Tokens, Patterns, and Lexemes
• Token
- A string of characters which logically belong together
- Classes of similar lexemes
l identifier, keywords, constants etc.
• Pattern
- A rule which describes a token
• Lexeme
- The sequence of characters matched by a pattern to form the token

Question sources

Compiler Design review questions · Question 16

Question 47 | Q01409

Which part of the compiler highly used the grammar concept?

A. Code optimization

B. Code generation

C. Parser

D. Lexical Analysis

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: Parser

PDF p. 9 explicitly states that parsers/syntax analyzers are generated for a particular grammar and CFGs specify programming-language syntax. This directly connects the parser to the grammar concept asked about.

Why each choice is right or wrong

A - Incorrect: Optimization is defined through removal of inefficiencies, not grammar-based recognition (p. 7).

B - Incorrect: Generation converts intermediate code to machine code (p. 7).

C - Correct: Parser generation and syntax specification are explicitly grammar-based (p. 9).

D - Incorrect: Lexical patterns classify tokens (p. 8); the cited grammar emphasis is the parser (p. 9).

Primary note proof

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 6-16

> Grammars
> • Every programming language has grammar rules
> • Parsers or syntax analyzers are generated for a particular grammar
> • CFG are used for syntax specification of programming languages
> Context Free Grammar (CFG)
> • Is denoted as G = (N, T , P, S)
> • N : finite set of non-terminals
> • T : finite set of terminals
> • S ∈ N: The start symbol
> • P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Compiler Design module | PDF page 5 | lines 12-18

> 2. Syntax / Hierarchical Analysis - Parsing
> • Tokens are grouped hierarchically into nested collections with collective meaning.
> • The result is generally a parse tree.
> • expressions, statements, declarations etc... are identified by using the results of lexical analysis.
> • Most syntactic errors in the source program are caught in this phase.
> • Syntactic rules of the source language are given via a Grammar.

Question sources

Compiler Design review questions · Question 17

Question 48 | Q01410

Which phase of the compiler checks the grammar of the programming?

A. Code Optimization

B. Semantic Analysis

C. Code Generation

D. Syntax Analysis

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Syntax Analysis

Syntax analysis groups tokens into nested constructs, catches syntactic errors, and uses source-language grammar rules (PDF p. 5). Therefore it checks grammatical program structure; semantic checks, optimization and target-code generation perform different work.

Why each choice is right or wrong

A - Incorrect: Optimization removes intermediate-code inefficiencies (p. 7).

B - Incorrect: Semantic analysis checks meaningful compatibility and types (p. 6).

C - Incorrect: Code generation emits machine code from intermediate code (p. 7).

D - Correct: Syntax analysis is explicitly governed by grammar rules.

Primary note proof

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Compiler Design module | PDF page 7 | lines 2-6

5. Code Optimization
Changes the IC by removing such inefficiencies
Improve the code
a. Improvement may be time, space, or power consumption.
It changes the structure of programs,

Question sources

Compiler Design review questions · Question 18

Question 49 | Q01411

Which listed data structure supplies declaration names, scopes and types to semantic analysis?

A. Yacc

B. Lex

C. Symbol Table

D. Type Checking

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: Symbol Table

A symbol table supplies the identities and attributes needed to resolve names and check types during semantic analysis. Type checking is an important activity using that information, but is not itself the requested data structure. Yacc and Lex generate parsers and scanners respectively. This narrower role makes the symbol table unique without denying that several components contribute to semantic processing.

Why each choice is right or wrong

A - Incorrect: Yacc is a parser generator.

B - Incorrect: Lex is a scanner generator.

C - Correct: The symbol table stores declaration and type metadata.

D - Incorrect: Type checking is an activity, not the information structure.

Primary note proof

Study-note section COMP19

> A symbol table maps names or symbol identities to attributes: type, scope, width, storage offset, parameter signature, constant information or linkage.

Study-note section COMP39

> Lex/flex generates scanners from patterns and actions. Yacc means Yet Another Compiler Compiler; Yacc/Bison generates parsers from grammar rules and semantic actions.

Study-note section COMP22

> A type checker verifies that operations are defined for their operands.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

> 3. Semantic Analysis
> • Certain checks are performed to make sure that the components of the program fit together meaningfully.
> • Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
> - Type checking of various programming language constructs is one of the most important tasks.
> • Stores type information in the symbol table or the syntax tree.
> - Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 14 | lines 2-15

• Type Errors:
- Improper or inconsistent operations during program execution

• Type-safety:
- Absence of type errors

Type Checking
• Semantic checks to enforce the type safety of the program
• Semantic Checks
- Static - done during compilation
- Dynamic - done during run-time

• Examples
- Unary and binary operators
- Number and type of arguments
- Return statement with return type
- Compatible assignment

Question sources

Compiler Design review questions · Question 19

Question 50 | Q01412

Which phase of the compiler is also known as Scanner?

A. Syntax Analysis

B. Lexical Analysis

C. Semantic Analysis

D. Code generation

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Lexical Analysis

The module labels lexical/linear analysis "scanning" (PDF p. 5). Its scanner reads characters into tokens (p. 8), so B names the phase associated with Scanner.

Why each choice is right or wrong

A - Incorrect: Syntax analysis is called parsing.

B - Correct: Lexical analysis is explicitly scanning.

C - Incorrect: Semantic analysis checks types/meaning (p. 6).

D - Incorrect: Code generation converts intermediate code to machine code (p. 7).

Primary note proof

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

1. Lexical / Linear Analysis (scanning)
- Scans the source code as a stream of characters
- Represent lexemes in the form of tokens as:
<token-name, attribute-value>
- Token
• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Question sources

Compiler Design review questions · Question 20

Question 51 | Q01413

Which phase of the compiler is also known as Parser?

A. Code Optimization

B. Semantic Analysis

C. Syntax Analysis

D. Lexical Analysis

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: Syntax Analysis

Syntax/hierarchical analysis is explicitly labeled Parsing (PDF p. 5); p. 9 equates parsers with syntax analyzers. Thus the phase known as Parser is syntax analysis.

Why each choice is right or wrong

A - Incorrect: Optimization improves intermediate code (p. 7).

B - Incorrect: Semantic analysis checks meaningful compatibility (p. 6).

C - Correct: Syntax analysis is expressly Parsing (pp. 5, 9).

D - Incorrect: Lexical analysis is scanning, preceding parsing.

Primary note proof

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

> 2. Syntax / Hierarchical Analysis - Parsing
> • Tokens are grouped hierarchically into nested collections with collective meaning.
> • The result is generally a parse tree.
> • expressions, statements, declarations etc... are identified by using the results of lexical analysis.
> • Most syntactic errors in the source program are caught in this phase.
> • Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 9 | lines 6-16

> Grammars
> • Every programming language has grammar rules
> • Parsers or syntax analyzers are generated for a particular grammar
> • CFG are used for syntax specification of programming languages
> Context Free Grammar (CFG)
> • Is denoted as G = (N, T , P, S)

• N : finite set of non-terminals
• T : finite set of terminals
• S ∈ N: The start symbol
• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Question sources

Compiler Design review questions · Question 21

Question 52 | Q01414

Which of the following parser is a top-down parser?

A. An LALR parser

B. A LR parser

C. Operator precedence parser

D. Recursive descent parser

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer D: Recursive descent.

COMP11 introduces recursive descent as a top-down implementation, normally with one procedure per nonterminal. It expands the start symbol toward the input. LR and LALR instead use bottom-up reductions, and operator-precedence parsing is also bottom-up. Therefore only recursive descent names the top-down family among these options.

Why each choice is right or wrong

A - Incorrect: LALR is an LR bottom-up family, using merged LR item states.

B - Incorrect: LR recognizes input through bottom-up shifts and reductions.

C - Incorrect: Operator-precedence parsing reduces expressions bottom-up rather than expanding productions top-down.

D - Correct: Recursive descent implements top-down parsing through procedures corresponding to nonterminals.

Primary note proof

Study-note section COMP11

Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using

lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP15

Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse.

Study-note section COMP-S01

Operator-precedence parsing is a restricted bottom-up expression-parsing method using a stack and precedence relations. With a suitable expression grammar, the stack retains unfinished surrounding expressions while nested parentheses are processed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

Parsing
• Process of constructing parse tree for a sentence generated by a given grammar.
• 2 types of parsers
- Top down parsing (predictive parsers)
• LL(1)
- Bottom up parsing (SR parsers)
• LR(1)

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
- the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Compiler Design review questions · Question 22

Question 53 | Q01415

Keywords are recognized in a compiler during -

A. the code generation

B. the data flow analysis

C. the lexical analysis of the program

D. the program parsing

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: the lexical analysis of the program

Keywords are listed among the token categories identified during lexical analysis (PDF pp. 5, 8). Recognizing a reserved spelling therefore occurs in the lexical classification of the program rather than later target generation or parsing.

Why each choice is right or wrong

A - Incorrect: Generation acts on intermediate code, not raw keyword spellings (p. 7).

B - Incorrect: Data-flow analysis is not identified as keyword recognition.

C - Correct: Keywords are explicitly lexical token classes (pp. 5, 8).

D - Incorrect: Parsing groups the already-produced tokens.

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

> 1. Lexical / Linear Analysis (scanning)
> - Scans the source code as a stream of characters
> - Represent lexemes in the form of tokens as:

<token-name, attribute-value>
- Token
- smallest meaningful element that a compiler understands.
- Eg.
- Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 8 | lines 19-24

- Classes of Tokens
- Identifiers: names chosen by the programmer
- Keywords: names already in the programming language
- Separators: punctuation characters
- Operators: symbols that operate on arguments and produce results
- Literals: numeric, textual literals

Question sources

Compiler Design review questions · Question 23

Question 54 | Q01417

Which names denote the graph whose vertices are basic blocks and edges are possible next-block control transfers? Select all that apply. In A, control graph abbreviates control-flow graph.

A. Control graph

B. DAG

C. Flow graph

D. Hamilton graph

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer A, C: Control graph; Flow graph

A control-flow graph represents basic blocks as vertices and possible execution transfers, including fall-through, as edges. With the stated abbreviation, both control graph and flow graph name that representation. An expression DAG instead represents shared computations and value dependencies. A Hamiltonian-path label concerns a graph property rather than the compiler representation of block successors. Thus A and C are both valid names.

Why each choice is right or wrong

A - Correct: The declared abbreviation denotes the specified graph.

B - Incorrect: An expression DAG records value dependencies, not block successors.

C - Correct: Flow graph conventionally names this representation.

D - Incorrect: Hamiltonian paths do not define the block-successor representation.

Primary note proof

Study-note section COMP28

> A basic block is a maximal straight-line instruction sequence entered at its beginning and exited at its end. Leaders are the first instruction, every jump target, and each instruction after a branch. Start a block at each leader and continue until the next leader.

Study-note section COMP28

> A control-flow graph has blocks as vertices and possible next-block transfers as edges, including fall-through. A node D dominates N when every entry-to-N path passes through D. For an edge N→H where H dominates N, H is a loop header of a natural loop. This structure supports loop analysis. An expression DAG records value dependencies, not arbitrary control flow.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 18 | lines 2-10

> Techniques
> • Common sub-expression elimination
> ● Repeated appearance computed previously
> • Strength reduction
> ● Replacement of expensive expressions with simple ones
> • Code movement
> ● Moving a block of code outside a loop
> • Dead code elimination
> ● Eliminated code statements that are either never executed or unreachable

Compiler Design module | PDF page 16 | lines 2-8

> Three Address Code
> • Is a generic form and can be implemented as:
> ● Quadruples
> ● Triples
> ● Indirect Triples
> ● Tree
> ● DAG

Question sources

Compiler Design review questions · Question 25

Question 55 | Q01420

In the compiler, the function of using intermediate code is:

A. to improve the register allocation

B. to increase the error reporting & recovery.

C. to make semantic analysis easier.

D. to increase the chances of re-using the machine-independent code optimizer in other compilers.

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer D: Reuse the machine-independent optimizer across compilers.

COMP26 explains that IR separates source analysis from target implementation, making optimizers and backends reusable. This directly supports reusing a machine-independent optimizer in different compilers. An IR can support allocation and analysis, but none of the other proposed improvements is its defining architectural function, nor does merely introducing IR guarantee them.

Why each choice is right or wrong

A - Incorrect: Allocation may analyze IR, but improved allocation is not the portability function identified here.

B - Incorrect: Error reporting and recovery are shared compiler services, not the defining purpose of introducing IR.

C - Incorrect: Semantic analysis normally establishes source correctness before lowering; IR is not introduced chiefly to simplify it.

D - Correct: Separating source and target through IR enables reuse of machine-independent optimization passes.

Primary note proof

Study-note section COMP26

> IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

> Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 3 | lines 25-29

Phases of a Compiler
• Analysis
● Machine Independent/Language Dependent
• Synthesis
● Machine Dependent/Language independent

Compiler Design module | PDF page 6 | lines 11-12

4. Intermediate Code Generation
Easy to produce and easy to translate to machine code

Compiler Design module | PDF page 7 | lines 2-6

5. Code Optimization
Changes the IC by removing such inefficiencies
Improve the code
a. Improvement may be time, space, or power consumption.
It changes the structure of programs,

Question sources

Compiler Design review questions · Question 28

Question 56 | Q01421

How many categories result when optimizations are divided solely by whether they depend on the target machine?

A. two types

B. three types

C. four types

D. five types

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer A: two types

The stated criterion divides optimizations into machine-dependent and machine-independent categories, giving two categories. Other taxonomies use different criteria, such as local, global and loop scope, and need not have the same count. Individual transformations are not additional categories under this binary division. Naming the criterion fixes the count without claiming a universal number of optimization types across all classifications.

Why each choice is right or wrong

A - Correct: The two categories are machine-dependent and machine-independent.

B - Incorrect: Three fits another possible scope taxonomy.

C - Incorrect: This binary division does not yield four categories.

D - Incorrect: This binary division does not yield five categories.

Primary note proof

Study-note section COMP32

> A loop-invariant expression has unchanged inputs during a loop. Hoist it to a preheader only when evaluation is safe on the paths where it will newly run. Induction variables change by a loop-invariant increment, such as i=i+1. For repeated t=4*i, maintain a running t increased by 4 when equivalent, reducing repeated multiplication.

Study-note section COMP32

> Loop fusion or jamming combines loops with compatible bounds and safe dependence order. Unrolling replicates a body to reduce branch overhead and expose parallelism; handle remaining iterations and increased code size. Loop-invariant code motion, fusion, unrolling and strength reduction are different transformations. A finite taxonomy needs a criterion: machine dependent/independent versus local/global/loop are different classifications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 2-6

> 5. Code Optimization
> Changes the IC by removing such inefficiencies
> Improve the code
> a. Improvement may be time, space, or power consumption.
> It changes the structure of programs,

Compiler Design module | PDF page 17 | lines 16-22

> Chapter 8
> Introduction to Code Optimization
> Goals of Code Optimization
> • Remove redundant code without changing the meaning of program
> • Executes faster
> • Efficient memory usage
> • Better performance

Question sources

Compiler Design review questions · Question 29

Question 57 | Q01422

The value of which variable is updated inside the loop by a loop-invariant value?

A. loop

B. strength

C. induction

D. invariable

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer C: An induction variable.

COMP32 defines induction variables as values changed by a loop-invariant increment, such as i=i+1. This matches an update inside a loop by an invariant amount. A loop-invariant value itself stays unchanged; strength is part of the term strength reduction, and loop alone names a construct rather than this variable class.

Why each choice is right or wrong

A - Incorrect: Loop names the repeated control construct rather than the variable category defined by invariant increments.

B - Incorrect: Strength refers to replacing costly operations, not to this variable classification.

C - Correct: An induction variable is updated by a loop-invariant increment during loop iterations.

D - Incorrect: An invariant value remains unchanged, unlike the updated variable described in the stem.

Primary note proof

Study-note section COMP32

> A loop-invariant expression has unchanged inputs during a loop. Hoist it to a preheader only when evaluation is safe on the paths where it will newly run. Induction variables change by a loop-invariant increment, such as i=i+1. For repeated t=4*i, maintain a running t increased by 4 when equivalent, reducing repeated multiplication.

Study-note section COMP32

> Loop fusion or jamming combines loops with compatible bounds and safe dependence order. Unrolling replicates a body to reduce branch overhead and expose parallelism; handle remaining iterations and increased code size. Loop-invariant code motion, fusion, unrolling and strength reduction are different transformations. A finite taxonomy needs a criterion: machine dependent/independent versus local/global/loop are different classifications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 18 | lines 2-10

Techniques
• Common sub-expression elimination
  • Repeated appearance computed previously
• Strength reduction
  • Replacement of expensive expressions with simple ones
• Code movement
  • Moving a block of code outside a loop
• Dead code elimination
  • Eliminated code statements that are either never executed or unreachable

Question sources

Compiler Design review questions · Question 30

Question 58 | Q01423

Which compiler runs on one machine and generates code for multiple machines?

A. Multipass compiler

B. Cross compiler

C. Optimizing compiler

D. Onepass compiler

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Cross compiler

A cross-compiler runs on platform A and generates executable code for another platform B (PDF p. 3). B matches the host/target distinction. The module's definition concerns a different target platform; it does not require one executable to support several targets simultaneously.

Why each choice is right or wrong

A - Incorrect: Multipass is not the host/target distinction described on p. 3.

B - Correct: Cross compiler is explicitly the platform-A to platform-B translator (p. 3).

C - Incorrect: Optimizing refers to improving code (p. 7), not the host/target relationship.

D - Incorrect: Onepass is not the documented host/target category.

Primary note proof

Study-note section COMP02

A preprocessor handles directives such as file inclusion and macros. An assembler translates assembly to object code. A linker combines object files and libraries, resolves external symbols and performs relocation. A loader maps an executable into memory and starts it. Compilation alone does not necessarily produce a ready executable.

Study-note section COMP02

A cross-compiler runs on a host platform and generates code for a different target. Multiple targets require appropriate backends, not a special definition of cross-compilation. A retargetable compiler supports changing target descriptions. Incremental compilation recompiles affected fragments or dependent units after edits; it is more than literally replacing changed lines. One-pass versus multipass counts traversals, not supported targets.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 3 | lines 20-24

• Cross-Compiler
● compiler that runs on platform (A) and generates executable code for another platform (B).
• Source-to-source Compiler
● compiler that translates source code of one programming language to another

Question sources

Compiler Design review questions · Question 31

Question 59 | Q01425

Which phenomenon happens when the non-terminal on the left side is repeated as the first symbol on the right side?

A. Left-most derivation

B. Left recursion

C. Left factoring

D. Left parsing

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: Immediate left recursion.

COMP12 writes immediate left recursion as A→Aα: the production's left-hand nonterminal occurs first on its right-hand side. This is precisely the pattern in the question. Leftmost derivation chooses which nonterminal to expand; left factoring extracts a shared prefix. Neither is the repeated-leftmost-self-reference property.

Why each choice is right or wrong

A - Incorrect: Leftmost derivation chooses the leftmost remaining nonterminal to rewrite, regardless of self-reference.

B - Correct: A production beginning with its own left-hand nonterminal is immediate left recursion.

C - Incorrect: Left factoring extracts common prefixes from alternatives and does not define recursive self-reference.

D - Incorrect: Left parsing is not the note's name for the production pattern described.

Primary note proof

Study-note section COMP12

Immediate left recursion has A→Aα. For A→Aα|β, where β does not start with A, introduce a new nonterminal A′ and use:
A→βA′
A′→αA′|ε
This preserves the generated language; tree shape changes, so semantic actions must preserve intended associativity. Indirect left recursion follows a cycle through other nonterminals and also needs removal for ordinary predictive descent.

Study-note section COMP12

Left factoring rewrites A→αβ|αγ using:
A→αA′
A′→β|γ
It postpones a choice until after the common prefix. It does not by itself remove ambiguity or guarantee LL(1). LR parsers can handle suitable left-recursive grammars; ordinary unmodified recursive descent can recurse forever before consuming input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 6-16

Grammars
• Every programming language has grammar rules
• Parsers or syntax analyzers are generated for a particular grammar
• CFG are used for syntax specification of programming languages
Context Free Grammar (CFG)
• Is denoted as G = (N, T , P, S)
• N : finite set of non-terminals
• T : finite set of terminals
• S ∈ N: The start symbol
• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Compiler Design module | PDF page 10 | lines 18-23

Top Down Parsing
• The parse tree is created top to bottom
• Starts from the start symbol and transform it to the input
Bottom Up Parsing

• Starts with the input symbols and tries to construct the parse tree up to the start symbol.
• One way of reducing a sentence is to follow the right most derivation in reverse

Question sources

Compiler Design review questions · Question 33

Question 60 | Q01428

The compiler can detect what type of errors?

A. neither logical nor grammatical error

B. logical errors only

C. grammatical errors only

D. both grammatical and logical errors

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: grammatical errors only

PDF p. 5 says syntax analysis catches syntactic errors, so grammar violations are detectable. PDF p. 6 adds semantic type mismatches; it does not claim detection of arbitrary mistakes in the intended algorithm. Among the offered grammatical-versus-logical choices, C fits if "logical" means wrong intended algorithm rather than all semantic errors.

Why each choice is right or wrong

A - Incorrect: Neither conflicts with the documented detection of syntactic errors.

B - Incorrect: Logical-only omits the explicitly detected grammatical errors.

C - Correct: Grammatical errors are expressly detected; the intended-algorithm interpretation of logical makes this the matching choice.

D - Incorrect: The module does not claim that general intended-algorithm errors are detected.

Primary note proof

Study-note section COMP08

A lexical error prevents forming a legal token, such as an invalid character or unterminated literal. A syntax error is an illegal token arrangement, such as a missing delimiter. A semantic error violates name, scope or type rules after grammatical recognition. A wrong algorithm can compile successfully; compilers do not detect all logical mistakes.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Compiler Design review questions · Question 36

Question 61 | Q01430

Which method merges the multiple loops into the single one?

A. Constant Folding

B. Loop rolling

C. Loop fusion or jamming

D. None of the above

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer C: Loop fusion, also called jamming.

COMP32 explicitly defines loop fusion or jamming as combining loops with compatible bounds and safe dependence order. That is the requested merging transformation. Constant folding evaluates constant expressions, while loop rolling does not name this fusion of separate loops. Fusion must preserve dependence order; it is not automatically legal for arbitrary loops.

Why each choice is right or wrong

A - Incorrect: Constant folding evaluates constant expressions at compile time; it does not combine loop bodies.

B - Incorrect: Loop rolling is not the note's named transformation for merging separate compatible loops.

C - Correct: Fusion or jamming combines compatible loops into one while preserving required dependence order.

D - Incorrect: The explicit fusion/jamming option already names the requested transformation.

Primary note proof

Study-note section COMP32

> A loop-invariant expression has unchanged inputs during a loop. Hoist it to a preheader only when evaluation is safe on the paths where it will newly run. Induction variables change by a loop-invariant increment, such as i=i+1. For repeated t=4*i, maintain a running t increased by 4 when equivalent, reducing repeated multiplication.

Study-note section COMP32

> Loop fusion or jamming combines loops with compatible bounds and safe dependence order. Unrolling replicates a body to reduce branch overhead and expose parallelism; handle remaining iterations and increased code size. Loop-invariant code motion, fusion, unrolling and strength reduction are different transformations. A finite taxonomy needs a criterion: machine dependent/independent versus local/global/loop are different classifications.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 18 | lines 2-10

> Techniques
> • Common sub-expression elimination
> • Repeated appearance computed previously
> • Strength reduction
> • Replacement of expensive expressions with simple ones
> • Code movement
> • Moving a block of code outside a loop
> • Dead code elimination
> • Eliminated code statements that are either never executed or unreachable

Question sources

Compiler Design review questions · Question 38

Question 62 | Q01431

Which parser is known as the shift-reduce parser?

A. Bottom-up parser

B. Top-down parser

C. Both Top-down and bottom-up

D. None of the Above

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer A: Bottom-up parser

PDF p. 10 names bottom-up parsing as SR parsing and describes reductions from the input to the start symbol. Interpreting SR as shift-reduce, A is the matching category; top-down is instead predictive.

Why each choice is right or wrong

A - Correct: Bottom-up is explicitly the SR category (p. 10).

B - Incorrect: Top-down is labeled predictive, not SR (p. 10).

C - Incorrect: Both collapses two categories the module distinguishes (p. 10).

D - Incorrect: The bottom-up category is present (p. 10), so none is false.

Primary note proof

Study-note section COMP15

> Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse. A handle is a production RHS occurrence whose reduction is the next step of that reverse derivation. Not every substring matching a RHS is a handle in the current context.

Study-note section COMP15

> Shift pushes the next input token and a state; reduce replaces a recognized RHS by its LHS and consults GOTO; accept recognizes completion; error reports an invalid situation. For A→XYZ, pop three grammar symbols and their states, expose state q, then push A and GOTO[q,A]. “Go” is not a separate standard shift-reduce ACTION, although GOTO is essential table machinery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

> Parsing
> • Process of constructing parse tree for a sentence generated by a given grammar.
> • 2 types of parsers
> ● Top down parsing (predictive parsers)
> • LL(1)
> ● Bottom up parsing (SR parsers)
> • LR(1)

Compiler Design module | PDF page 10 | lines 18-23

> Top Down Parsing
> • The parse tree is created top to bottom

• Starts from the start symbol and transform it to the input
Bottom Up Parsing
• Starts with the input symbols and tries to construct the parse tree up to the start symbol.
• One way of reducing a sentence is to follow the right most derivation in reverse

Question sources

Compiler Design review questions · Question 39

Question 63 | Q01433

The full form of YACC is:

A. Yet Another Computer Computer

B. Yet Another Computer Compiler

C. Yet Another Compiler Computer

D. Yet Another Compiler Compiler

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Yet Another Compiler Compiler.

COMP39 expands Yacc as Yet Another Compiler Compiler and describes Yacc/Bison as generating parsers from grammar rules and semantic actions. The other expansions interchange computer and compiler and do not match the note. The name refers to a parser generator, distinct from Lex/flex scanner generation.

Why each choice is right or wrong

A - Incorrect: Computer Computer replaces both required occurrences of Compiler and is not the stated expansion.

B - Incorrect: The first noun must be Compiler, not Computer.

C - Incorrect: The final noun must also be Compiler, not Computer.

D - Correct: This matches the exact Yacc expansion given in COMP39.

Primary note proof

Study-note section COMP39

Panic-mode recovery skips input until a synchronizing token such as ; or } permits parsing to resume. Phrase-level recovery makes a local insertion/deletion. Report the location and cause while avoiding cascades of misleading diagnostics. Lexical, syntax and semantic phases each detect their own errors; error handling is a shared service.

Study-note section COMP39

Lex/flex generates scanners from patterns and actions. Yacc means Yet Another Compiler Compiler; Yacc/Bison generates parsers from grammar rules and semantic actions. Typical specification files have definitions, %% rules, and %% user code. Classic Yacc uses LALR; Bison supports additional LR and generalized modes. Scanner generation does not replace the CFG parser.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 6-16

Grammars
• Every programming language has grammar rules
• Parsers or syntax analyzers are generated for a particular grammar
• CFG are used for syntax specification of programming languages
Context Free Grammar (CFG)
• Is denoted as G = (N, T , P, S)
• N : finite set of non-terminals
• T : finite set of terminals
• S ∈ N: The start symbol
• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Question sources

Compiler Design review questions · Question 41

Question 64 | Q01434

What type of conflicts can occur in the shift-reduce parsing? Select the most complete choice.

A. reduce/reduce

B. shift/reduce

C. Both shift/reduce and reduce/reduce

D. None of the above

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer C: Both shift/reduce and reduce/reduce

COMP17 distinguishes a shift/reduce conflict, where shifting competes with reduction, from reduce/reduce, where two reductions compete. Both can arise in shift-reduce table construction. The combined option therefore answers the question completely; either individual conflict is a valid example but omits the other named kind.

Why each choice is right or wrong

A - Incorrect: True as one component, but incomplete as the requested full selection. Two competing reductions form a reduce/reduce conflict, one of the listed possible types.

B - Incorrect: True as one component, but incomplete as the requested full selection. A shift competing with a reduction forms the other standard conflict type.

C - Correct: This aggregate includes both conflict types explicitly defined in COMP17.

D - Incorrect: Both listed kinds can occur, so denying every listed type is false.

Primary note proof

Study-note section COMP17

> SLR(1) uses LR(0) states and FOLLOW(A) for reduction by a completed A production. Canonical LR(1) attaches context-specific lookahead to items. LALR(1) merges canonical states with the same LR(0) core, combining lookaheads; merging can introduce reduce/reduce conflicts. The standard grammar-class hierarchy is LR(0)⊆SLR(1)⊆LALR(1)⊆LR(1), with increasing capability and often table size.

Study-note section COMP17

> A shift/reduce conflict offers both consuming a token and reducing; reduce/reduce offers two reductions. An SLR or LALR conflict does not by itself prove the grammar ambiguous. Canonical LR(1) handles LR(1) grammars, not every unambiguous CFG. Precedence and associativity can resolve suitable expression conflicts; changing parser names alone does not repair all ambiguity.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

> Parsing
> • Process of constructing parse tree for a sentence generated by a given grammar.
> • 2 types of parsers
> ● Top down parsing (predictive parsers)
> • LL(1)
> ● Bottom up parsing (SR parsers)
> • LR(1)

Compiler Design module | PDF page 11 | lines 2-17

> LL(1) Grammar
> • L - left to right
> • L - left most derivation
> • 1 - number of look ahead
> • First( ) and Follow( )
> ● the first terminal in a string and the terminal that follows a variable respectively.
> LR Parsing
> • LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
> Types of LR Parsers

• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Compiler Design review questions · Question 42

Question 65 | Q01436

Which algorithm invokes a function GETREG()?

A. Code motion algorithm

B. Code optimization algorithm

C. Intermediate Code

D. Code generation algorithm

Hint

Match stored information to its lifetime, access path and calling-convention role; distinguish compile-time organization from run-time storage.

Answer D: A code-generation algorithm.

COMP35 directly associates getReg with a simple code-generation algorithm. It selects operand/result registers for a TAC instruction before loads and the target operation are emitted and descriptors are updated. Code motion and general optimization serve other transformation purposes; intermediate code is the input representation here rather than the algorithm making the register choice.

Why each choice is right or wrong

A - Incorrect: Code motion relocates computations; it is not the described getReg-driven emission algorithm.

B - Incorrect: Optimization is broader, but COMP35 specifically locates getReg in the target code-generation algorithm.

C - Incorrect: Intermediate code supplies instructions to the generator; it is not the algorithm invoking the register selector.

D - Correct: The note explicitly describes getReg as part of simple target code generation.

Primary note proof

Study-note section COMP35

A register descriptor records which current values are held in a register. An address descriptor for a name records all current locations containing its value, possibly memory and multiple registers. A memory copy can become stale after a register-only assignment. These are code-generator bookkeeping structures, not source-language types.

Study-note section COMP35

A simple code-generation algorithm calls getReg for a TAC instruction to select operand/result registers, emits necessary loads, emits the operation, then updates descriptors. Prefer an already suitable register or a reusable dead operand; otherwise choose a spill victim and store any live value lacking another current copy. At a block exit, preserve values needed by successors.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Compiler Design module | PDF page 18 | lines 11-22

Register Allocation
• Registers hold values
• Example
- a = c + d
- e = a + b
- f = e - 1

• With the assumption that a and e die after use
• Temporary a can be reused after e=a+b, same wz a
• Can allocate a,e and f all to one register(r1)
- r1 = r2 + r3
- r1 = r1 + r4
- r1 = r1 - 1

Question sources

Compiler Design review questions · Question 44

Question 66 | Q01438

The execution time of the code depends on? Select the most complete choice.

A. the usage of machine idioms

B. the way the registers are used

C. the orders in which the computations are performed

D. All of the mentioned

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: All of the mentioned

COMP34 states that machine idioms, register availability and instruction order affect execution time. These correspond to all three concrete choices. Better register use can reduce loads/spills; target idioms affect instruction cost, and legal scheduling changes execution efficiency. The aggregate is therefore the complete response rather than any one factor in isolation.

Why each choice is right or wrong

A - Incorrect: True as one component, but incomplete as the requested full selection. Choosing target machine idioms changes the operations emitted and their target costs.

B - Incorrect: True as one component, but incomplete as the requested full selection. Register use affects memory traffic and spills, which can change execution time.

C - Incorrect: True as one component, but incomplete as the requested full selection. The scheduling of computations affects execution efficiency while dependencies constrain legal orders.

D - Correct: COMP34 explicitly includes all three factors, so the aggregate is the complete answer.

Primary note proof

Study-note section COMP34

> Code generation lowers checked IR to a target form. Instruction selection chooses target operations; register allocation chooses where values reside; scheduling orders instructions while respecting dependencies; storage layout assigns offsets and follows ABI requirements. Register assignment names actual physical registers after allocation decisions.

Study-note section COMP34

> For t=a+b, a simple target could emit LOAD R1,a; LOAD R2,b; ADD R3,R1,R2; STORE t,R3. Other targets may combine loads with arithmetic or keep t in a register. Machine idioms, register availability and instruction order affect execution time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 7-11

> 6. Code Generation
> Converts intermediate code to machine code.
> Must handle all aspects of machine architecture
> Storage allocation decisions are made
> a. Register allocation and assignment

Compiler Design module | PDF page 18 | lines 11-22

Register Allocation
• Registers hold values
• Example
- a = c + d
- e = a + b
- f = e - 1

• With the assumption that a and e die after use
• Temporary a can be reused after e=a+b, same wz a
• Can allocate a,e and f all to one register(r1)
- r1 = r2 + r3
- r1 = r1 + r4
- r1 = r1 - 1

Compiler Design module | PDF page 17 | lines 16-22

Chapter 8
Introduction to Code Optimization
Goals of Code Optimization
• Remove redundant code without changing the meaning of program
• Executes faster
• Efficient memory usage
• Better performance

Question sources

Compiler Design review questions · Question 46

Question 67 | Q01439

Which is the most specific named method for examining a short instruction window and replacing jump L1 with jump L2 when L1 immediately jumps to L2?

A. Latter optimization technique

B. Peephole optimization technique

C. Local optimization technique

D. Code optimization technique

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer B: Peephole optimization technique

Peephole optimization examines a short instruction window and replaces it with an equivalent, preferable sequence. The described jump-chain rewrite is a direct example. Local optimization and code optimization are broader classifications that may include the transformation, but do not name the specific short-window method requested. Latter optimization is not a method defined in the supplied note.

Why each choice is right or wrong

A - Incorrect: Latter optimization does not name the specified method.

B - Correct: Peephole explicitly performs short-window jump-chain rewrites.

C - Incorrect: Local optimization is broader than the requested method.

D - Incorrect: Code optimization is broader than the requested method.

Primary note proof

Study-note section COMP31

> Peephole optimization examines short instruction windows. It can remove redundant loads/stores, replace machine idioms and shorten jump chains: jump L1 where L1 immediately jumps L2 can become jump L2.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 18 | lines 23-34

> Peephole Optimization
> • Transforming to optimal sequence of instructions
> Common Techniques:
> • Elimination of redundant loads and stores
> - Eg.
> - r2 = r1 + 5
> - I = r2
> - r3 = I
> - r4 = r3 * 3
>
> • Constant folding
> - Eg.
> - R2 = 3 * 2

Compiler Design module | PDF page 18 | lines 2-10

> Techniques
> • Common sub-expression elimination
> - Repeated appearance computed previously
>
> • Strength reduction
> - Replacement of expensive expressions with simple ones
>
> • Code movement
> - Moving a block of code outside a loop
>
> • Dead code elimination
> - Eliminated code statements that are either never executed or unreachable

Question sources

Compiler Design review questions · Question 47

Question 68 | Q01440

Which of the following term is used to keep track of the location where the current values of the name are stored?

A. Register descriptor

B. Address descriptor

C. Allocation descriptor

D. Flag register

Hint

Match stored information to its lifetime, access path and calling-convention role; distinguish compile-time organization from run-time storage.

Answer B: An address descriptor.

COMP35 defines an address descriptor for a name as recording all current locations that contain its value, including memory and registers. That matches tracking where a name's current value is stored. A register descriptor instead records values held in a particular register, reversing the direction of the association.

Why each choice is right or wrong

A - Incorrect: A register descriptor maps a register to its current values rather than a name to all locations.

B - Correct: An address descriptor records the current memory and register locations holding a name's value.

C - Incorrect: Allocation descriptor is not the note's defined bookkeeping structure for name-to-location tracking.

D - Incorrect: A flag register stores machine status flags rather than all locations of a named value.

Primary note proof

Study-note section COMP35

> A register descriptor records which current values are held in a register. An address descriptor for a name records all current locations containing its value, possibly memory and multiple registers. A memory copy can become stale after a register-only assignment.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 18 | lines 11-22

> Register Allocation
> • Registers hold values
> • Example
> - a = c + d
> - e = a + b

- f = e - 1
• With the assumption that a and e die after use
• Temporary a can be reused after e=a+b, same wz a
• Can allocate a,e and f all to one register(r1)
- r1 = r2 + r3
- r1 = r1 + r4
- r1 = r1 - 1

Question sources

Compiler Design review questions · Question 48

Question 69 | Q01441

In a nontrivial expression AST whose leaves are operands, which complete category of nodes carries operator symbols, including operators in nested subexpressions?

A. Root

B. Interior nodes

C. Leaves

D. Nodes

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Interior nodes

In an expression AST, leaves carry operands such as identifiers and constants, while nonleaf nodes carry the operators that combine child expressions. A nontrivial tree’s root can also be an operator, but naming only the root omits nested operators. Naming all nodes includes operand leaves. The complete category requested is therefore the interior nodes, which includes an operator root and any other nonleaf operator nodes.

Why each choice is right or wrong

A - Incorrect: The root alone omits operators in nested subexpressions.

B - Correct: Interior nodes are the complete nonleaf category carrying operators in this expression AST.

C - Incorrect: Leaves carry operands rather than combining operators.

D - Incorrect: All nodes includes operand leaves and is too broad.

Primary note proof

Study-note section COMP09

A context-free grammar G=(N,T,P,S) has nonterminals N, terminals T, productions P and start S. A production A→α rewrites one nonterminal. A sentential form is any string of terminals/nonterminals derivable from S; a sentence contains only terminals. A leftmost derivation rewrites the leftmost remaining nonterminal; a rightmost one rewrites the rightmost.

Study-note section COMP09

A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input. An AST omits grammar scaffolding and punctuation; expression operands are leaves and operators internal nodes. The parser can build either representation, depending on implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 26-28

Functions for Syntax Tree Nodes
• mknode (op, left, right)
- Creates an operator node with label op &

Compiler Design module | PDF page 13 | lines 2-14

- Two fields containing pointers to left and right
• mkleaf(id, entry)
- Creates an identifier node with label id &
- A field containing entry, ptr to symbol table entry for the identifier
• mkleaf(num, val)
- Create a number node with label num &
- A field containing val, the value for the number

Syntax tree for expression a-4+c
• P1=mkleaf(id,entrya);
• P2=mkleaf(num, 4);
• P3=mknode('-',p1,p2);
• P4=mkleaf(id,entryc);
• P5=mknode('+',p3,p4);

Question sources

Compiler Design review questions · Question 49

Question 70 | Q01443

Which of the following is used in various stages or phases of the compiler?

A. Records

B. Program

C. Symbol Table

D. Table

Hint

Ask whether the required information is a declaration's identity and attributes, program structure or the ordered source tokens.

Answer C: Symbol Table

The module stores semantic type information in the symbol table (PDF p. 6), puts identifier-entry pointers in syntax-tree leaves (p. 13), and enters names/types during declarations (p. 17). These uses support the symbol table as a shared structure across compiler work.

Why each choice is right or wrong

A - Incorrect: Records is only a generic storage term, not the shared named structure in these passages.

B - Incorrect: Program is the object being compiled, not the metadata table.

C - Correct: Symbol Table is explicitly used in several contexts (pp. 6, 13, 17).

D - Incorrect: Table is nonspecific; Symbol Table is the documented structure.

Primary note proof

Study-note section COMP19

> A symbol table maps names or symbol identities to attributes: type, scope, width, storage offset, parameter signature, constant information or linkage.

Study-note section COMP03

> Symbol-table management and error reporting support multiple phases rather than forming one mandatory seventh final phase.

Study-note section COMP19

> Scanning may intern a spelling first; semantic analysis determines the declaration and type.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

> 3. Semantic Analysis
> • Certain checks are performed to make sure that the components of the program fit together meaningfully.
> • Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
> - Type checking of various programming language constructs is one of the most important tasks.
> • Stores type information in the symbol table or the syntax tree.
> - Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 13 | lines 2-14

- Two fields containing pointers to left and right
• mkleaf(id, entry)
- Creates an identifier node with label id &
- A field containing entry, ptr to symbol table entry for the identifier
• mkleaf(num, val)
- Create a number node with label num &
- A field containing val, the value for the number

Syntax tree for expression a-4+c
• P1=mkleaf(id,entrya);
• P2=mkleaf(num, 4);
• P3=mknode('-',p1,p2);
• P4=mkleaf(id,entryc);
• P5=mknode('+',p3,p4);

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float
id.width=4
offset = offset +id.width { offset = 6 }

Question sources

Compiler Design review questions · Question 51

Question 71 | Q01444

Which of the following structure has four fields?

A. Parse tree

B. Triples

C. Indirect Triples

D. Quadruples

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer D: Quadruples

PDF p16 explicitly divides a quadruple into operator, arg1, arg2, and result: four fields. It contrasts triples with three fields.

Why each choice is right or wrong

A - Incorrect: A parse tree is a hierarchical representation, not the four-field instruction defined on p16.

B - Incorrect: P16 explicitly gives triples three fields.

C - Incorrect: P16 describes indirect triples through pointers, not a fourth result field.

D - Correct: P16 explicitly lists the four quadruple fields.

Primary note proof

Study-note section COMP26

> IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

> Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 16 | lines 13-23

> Three Address Code
> • Quadruples:
> - Each instruction is divided into four fields
> - Operator, arg1, arg2, and result
> • Triples:
> - Has three fields
> - Operator, arg1 and arg2
> • DAG and Tree
> - Similar presentation of expression to triples
> • Indirect Triples
> - Uses pointers instead of position to store results

Question sources

Compiler Design review questions · Question 52

Question 72 | Q01445

What is the conventional name for the compact expression representation with operand leaves and operator interior nodes?

A. Syntax tree

B. Parser tree

C. Structured tree

D. Sematic tree

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Syntax tree

The conventional expression AST places operands at leaves and operators at interior nodes, removing grammar scaffolding and punctuation. A parse tree instead records grammar productions: its internal nodes are nonterminals, and its yield includes the terminals of the source phrase. The other two names do not identify the conventional expression representation in this comparison. The syntax-tree choice supplies the specific structure described by the question.

Why each choice is right or wrong

A - Correct: An expression syntax tree places operands at leaves and operators at nonleaf nodes.

B - Incorrect: A parse tree's internal labels are grammar nonterminals, not generally expression operators.

C - Incorrect: This generic phrase does not specify the conventional expression AST representation requested.

D - Incorrect: Semantic annotation can be attached to representations, but this is not the conventional name for the described AST.

Primary note proof

Study-note section COMP09

> A context-free grammar G=(N,T,P,S) has nonterminals N, terminals T, productions P and start S. A production A→α rewrites one nonterminal. A sentential form is any string of terminals/nonterminals derivable from S; a sentence contains only terminals. A leftmost derivation rewrites the leftmost remaining nonterminal; a rightmost one rewrites the rightmost.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input. An AST omits grammar scaffolding and punctuation; expression operands are leaves and operators internal nodes. The parser can build either representation, depending on implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 26-28

Functions for Syntax Tree Nodes
• mknode (op, left, right)
- Creates an operator node with label op &

Compiler Design module | PDF page 13 | lines 2-14

- Two fields containing pointers to left and right
• mkleaf(id, entry)
- Creates an identifier node with label id &
- A field containing entry, ptr to symbol table entry for the identifier
• mkleaf(num, val)
- Create a number node with label num &
- A field containing val, the value for the number

Syntax tree for expression a-4+c
• P1=mkleaf(id,entrya);
• P2=mkleaf(num, 4);
• P3=mknode('-',p1,p2);
• P4=mkleaf(id,entryc);
• P5=mknode('+',p3,p4);

Compiler Design module | PDF page 9 | lines 22-27

Derivation Trees
• Derivations can be displayed as trees
• Internal nodes of the tree are all non-terminals
• Leaves are all terminals
• The yield of a derivation tree is the list of the labels of all the leaves read from left to right.

Question sources

Compiler Design review questions · Question 53

Question 73 | Q01446

Which statement is an abstract form of intermediate code?

A. 3- address

B. 2-address

C. address

D. Intermediate code

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer A: 3- address

PDF pp15-16 defines three-address code as a generic intermediate representation, X = Y op Z. Thus A names the supplied abstract statement form; D only repeats the general category.

Why each choice is right or wrong

A - Correct: P15 gives precisely this statement form.

B - Incorrect: No two-address intermediate statement form is supplied; it is not the form defined on p15.

C - Incorrect: A bare address is not the statement X = Y op Z.

D - Incorrect: This names the category rather than the particular statement form requested.

Primary note proof

Study-note section COMP26

> IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

> Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 15 | lines 13-21

> Three Address Code
> • Is a sequence of statements of the form
> - X = Y op Z
> - X,Y and Z are names, constants or compiler generated temporaries
> - Op is operator (arithmetic, logical)
>
> • Example:
> - a = b + c , x = -y , if a > b goto L1
>
> • LHS is the target
> • RHS has at most two sources and one operator

Compiler Design module | PDF page 16 | lines 2-8

> Three Address Code
> • Is a generic form and can be implemented as:

- Quadruples
- Triples
- Indirect Triples
- Tree
- DAG

Question sources

Compiler Design review questions · Question 54

Question 74 | Q01447

Which mapping is described by the implementation of the syntax-directed translator?

A. Parse table

B. Input

C. Output

D. Input-Output

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer D: An input-to-output mapping.

COMP25 distinguishes translation from recognition: semantic actions compute an output, such as values, ASTs or code, from an input program. Accordingly the implemented translator defines an input-to-output mapping. A parse table may drive an implementation, but neither the table nor either endpoint alone names that mapping.

Why each choice is right or wrong

A - Incorrect: A parse table can guide recognition; it is not the translation mapping itself.

B - Incorrect: Input is only the domain side and omits the translated result.

C - Incorrect: Output is only the result side and omits the mapping from source inputs.

D - Correct: Syntax-directed translation relates each processed input to its computed translation output.

Primary note proof

Study-note section COMP25

A syntax-directed translation scheme embeds executable actions in productions; the actions run while translating a source program, not while generating the compiler. Attributes may compute types, values, AST pointers or generated code. The scheme defines an input-to-output translation, not merely an input language.

Study-note section COMP25

Create an identifier leaf with mkleaf(id,entry), a number with mkleaf(num,value), and an operator node with mknode(op,left,right). For a−4+c, construct +(−(a,4),c). Postorder traversal prints postfix: a 4 − c +. A stack evaluator pushes operands and pops right then left operands for an operator. Infix-to-postfix is useful for stack machines but not mandatory for all compilers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 4-16

Syntax Directed Translation
• Attaching actions to the grammar rules(productions).
• Actions are executed during the compilation
- Not during the generation of the compiler
• Actions are executed according to the parsing mechanism.
Syntax Directed Definitions
• Is a generalization of a context free grammar
• Is a CFG with attributes and rules
• Attributes are associated with grammar symbols and rules with productions
• Attributes may be:
- Numbers
- Types
- Strings etc

Compiler Design module | PDF page 12 | lines 17-25

Syntax Directed Definition- Example
• Production Semantic Rules
• L → E return print(E.val)
• E → E1 + T E.val = E1.val + T.val
• E → T E.val = T.val
• T → T1 * F T.val = T1.val * F.val
• T → F T.val = F.val
• T → (E) F.val = E.val
• F → digit F.val = digit.lexval

Question sources

Compiler Design review questions · Question 55

Question 75 | Q01448

Which transition function moves the dot over a grammar symbol and takes closure to build successor states in the canonical LR(0) item collection?

A. FIRST

B. GOTO

C. COMPUTE

D. FOLLOW

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: GOTO

GOTO(I,X) advances the dot over X in eligible items and then takes closure, producing a successor item state. Repeating this over grammar symbols builds the canonical collection. The collection is the resulting set of item sets, not the function itself. FIRST and FOLLOW compute terminal sets for different parsing purposes; COMPUTE does not name the defined LR successor operation.

Why each choice is right or wrong

A - Incorrect: FIRST finds possible beginning terminals.

B - Correct: GOTO produces the successor item state.

C - Incorrect: COMPUTE is not the defined successor-state function.

D - Incorrect: FOLLOW finds possible following terminals.

Primary note proof

Study-note section COMP16

> Augment the grammar with S′→S. An LR(0) item A→α·β records progress through a production. CLOSURE(I) repeatedly adds B→·γ whenever an item has its dot before nonterminal B and B→γ is a production.

Study-note section COMP16

> GOTO(I,X) moves the dot past X in every eligible item and takes closure. Start with CLOSURE({S′→·S}); repeatedly compute nonempty GOTO sets for all grammar symbols until no new state appears. The resulting set of item sets is the canonical collection, not GOTO itself.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 11 | lines 2-17

> LL(1) Grammar
> • L - left to right
> • L - left most derivation

• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Compiler Design review questions · Question 56

Question 76 | Q01449

Which of the following option is not a function of the shift-reduce parser?

A. Reduce

B. Accept

C. Go

D. Shift

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer C: “Go” is not a standard shift-reduce ACTION.

COMP15 lists shift, reduce, accept and error as standard shift-reduce actions. GOTO is essential after a reduction but is separate table machinery, not a distinct action called Go. Reading function as the standard action category yields the offered Go choice; the three other choices are explicitly standard actions.

Why each choice is right or wrong

A - Incorrect: Reduce is a standard action that replaces a recognized right-hand side by its left-hand nonterminal.

B - Incorrect: Accept is a standard action indicating successful completion of recognition.

C - Correct: Go is not one of the standard ACTION names; GOTO is separate transition-table machinery.

D - Incorrect: Shift is a standard action that consumes input and pushes a token with its state.

Primary note proof

Study-note section COMP15

Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse. A handle is a production RHS occurrence whose reduction is the next step of that reverse derivation. Not every substring matching a RHS is a handle in the current context.

Study-note section COMP15

Shift pushes the next input token and a state; reduce replaces a recognized RHS by its LHS and consults GOTO; accept recognizes completion; error reports an invalid situation. For A→XYZ, pop three grammar symbols and their states, expose state q, then push A and GOTO[q,A]. “Go” is not a separate standard shift-reduce ACTION, although GOTO is essential table machinery.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

Parsing
• Process of constructing parse tree for a sentence generated by a given grammar.
• 2 types of parsers
- Top down parsing (predictive parsers)
• LL(1)
- Bottom up parsing (SR parsers)
• LR(1)

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
- the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Compiler Design review questions · Question 57

Question 77 | Q01450

Which listed grammar classes guarantee no adjacent nonterminals on a production right-hand side? Select all that apply. Regular means consistently right-linear; irregular means arbitrary unrestricted.

A. Arbitrary unrestricted grammar

B. Right-linear regular grammar

C. Operator grammar

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B, C: Right-linear regular grammar; Operator grammar

An operator grammar explicitly forbids adjacent nonterminals on a right-hand side. A consistently right-linear regular grammar has at most one nonterminal in such a rule, so it also guarantees the restriction. Arbitrary unrestricted grammars have no comparable adjacency guarantee. The property is therefore shared by the two specified classes and cannot be used to identify only operator grammar.

Why each choice is right or wrong

A - Incorrect: An unrestricted grammar can contain adjacent nonterminals.

B - Correct: A right-linear regular rule has at most one nonterminal.

C - Correct: Operator grammars forbid adjacent nonterminals by definition.

Primary note proof

Study-note section COMP18

> LL, LR and recursive-descent parsers can all handle nested parentheses for suitable grammars. An explicit or implicit stack, not the LR label alone, supports unbounded nesting. Both LL and LR can use parsing tables. Deterministic LL/LR grammars must be unambiguous; a general ambiguous grammar requires rewriting, disambiguation policy or a generalized method such as GLR.

Study-note section COMP18

> An operator grammar conventionally forbids adjacent nonterminals on a RHS and usually ε productions. Operator-precedence parsing additionally requires conflict-free precedence relations. Ordinary right-linear regular grammars also lack adjacent nonterminals, so that property alone does not uniquely identify the operator-grammar category. Regular patterns are chiefly lexical specifications; CFGs chiefly describe nested syntax.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 6-16

Grammars
• Every programming language has grammar rules
• Parsers or syntax analyzers are generated for a particular grammar
• CFG are used for syntax specification of programming languages
Context Free Grammar (CFG)
• Is denoted as G = (N, T , P, S)
• N : finite set of non-terminals
• T : finite set of terminals
• S ∈ N: The start symbol
• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Question sources

Compiler Design review questions · Question 58

Question 78 | Q01488

In an implementation where the scanner interns identifier spellings in a symbol table before returning their tokens, which phase first interacts with that table?

A. Syntax analysis

B. Parsing

C. Scanning

D. Semantic analysis

Hint

Ask whether the required information is a declaration’s identity and attributes, program structure or the ordered source tokens.

Answer C: Scanning

Interning associates an identifier spelling with a table entry while scanning its characters and returning the identifier token. Under this explicit implementation convention, scanning is the first phase to interact with the table. Parsing and semantic analysis use or enrich this information later; semantic analysis resolves declaration identity and type. This establishes an ordering for the stated implementation rather than a universal compiler requirement.

Why each choice is right or wrong

A - Incorrect: Syntax analysis receives tokens after this scanner has interned identifiers.

B - Incorrect: Parsing is syntax analysis and occurs after the stated interning.

C - Correct: Scanning performs the stipulated initial interning.

D - Incorrect: Semantic analysis enriches name/type information after that initial interaction.

Primary note proof

Study-note section COMP19

Scanning may intern a spelling first; semantic analysis determines the declaration and type. Implementations differ, so first interaction is conventional rather than universal.

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

1. Lexical / Linear Analysis (scanning)
- Scans the source code as a stream of characters
- Represent lexemes in the form of tokens as:
<token-name, attribute-value>
- Token
• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Mock exam 1 · Question 4

Question 79 | Q01489

What is the name given to the sequence of alphabets that are identified as a token?

A. Lexeme

B. Prefix

C. Pattern

D. Grammar

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Lexeme

PDF p8 calls the sequence of characters matched by a pattern to form a token a lexeme. Interpreting alphabets as characters makes A exact.

Why each choice is right or wrong

A - Correct: Matches the p8 definition.

B - Incorrect: Prefix is not the defined name for the matched sequence.

C - Incorrect: P8 defines a pattern as a rule, not the matched sequence.

D - Incorrect: P9 uses grammar for syntactic production rules, not a token occurrence.

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 10-18

> Tokens, Patterns, and Lexemes
> • Token
> ● A string of characters which logically belong together
> ● Classes of similar lexemes
> l identifier, keywords, constants etc.
> • Pattern
> ● A rule which describes a token
> • Lexeme
> ● The sequence of characters matched by a pattern to form the token

Question sources

Mock exam 1 · Question 5

Question 80 | Q01490

In a compiler where semantic analysis annotates an existing AST with resolved declarations and checked types, what is its resulting program representation?

A. Language constructs

B. Unannotated concrete syntax tree

C. AST annotated with resolved declarations and checked types

D. Token

Hint

Ask whether the required information is a declaration's identity and attributes, program structure or the ordered source tokens.

Answer C: AST annotated with resolved declarations and checked types

The stated implementation resolves names and checks types on an existing abstract syntax tree, so its result is that AST with semantic annotations. An unannotated syntax tree omits the newly checked information; language constructs is too generic to identify the representation. Tokens are the scanner's earlier output. Other compilers may produce typed IR, but the question specifies the retained-tree implementation.

Why each choice is right or wrong

A - Incorrect: Language constructs does not identify the specified annotated representation.

B - Incorrect: An unannotated concrete syntax tree is not the specified result.

C - Correct: The result is the AST enriched with checked semantic information.

D - Incorrect: Tokens precede syntax and semantic analysis.

Primary note proof

Study-note section COMP21

> Static checks occur before execution: type compatibility, duplicate declarations within a restricted scope, valid names, argument counts, return compatibility and control-flow placement. A break is valid only in constructs allowed by the language; in C/C++ it can exit a loop or switch. A compiler may annotate an AST or create typed IR; there is no mandatory uniquely named output tree.

Study-note section COMP21

> Dynamic checks depend on runtime values, such as some bounds and null checks. Static analysis can prove some accesses safe or faulty; "all memory errors cannot be statically checked" is too strong. Unreachable code is not universally a semantic error: some languages reject it in specified cases, others merely warn or optimize it away.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Question sources

Mock exam 1 · Question 6

Question 81 | SM0026

For const int x=8; const int y=16; if(x>y){int z=x+y;}, which techniques directly apply to evaluating the condition from known constants and removing its unreachable branch? Select all that apply. Distinguish constant evaluation from algebraic identity simplification.

A. Arithmetic simplification

B. Dead code elimination

C. Code motion

D. Compile time evaluation

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer B, D: Dead code elimination; Compile time evaluation

With x=8 and y=16, the condition x>y evaluates to false during compilation. The branch body is consequently unreachable and can be removed. These are compile-time evaluation and dead-code elimination, respectively. No computation needs to move to another program location, and the decisive evaluation is not an algebraic identity such as x+0→x. Both direct transformations belong in the answer.

Why each choice is right or wrong

A - Incorrect: No algebraic identity rewrite is the decisive step.

B - Correct: The constant-false branch is unreachable and removable.

C - Incorrect: Code motion does not describe evaluating or deleting the branch.

D - Correct: Known constants permit compile-time evaluation of the condition.

Primary note proof

Study-note section COMP30

> For const int x=8,y=16; if(x>y){int z=x+y;}, propagate the constants, fold 8>16 to false, then remove the unreachable branch. Both compile-time evaluation and dead-code elimination apply.

Study-note section COMP31

> Algebraic simplification removes identities when valid: integer x+0→x.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 18 | lines 2-10

> Techniques
> • Common sub-expression elimination
> ● Repeated appearance computed previously
> • Strength reduction
> ● Replacement of expensive expressions with simple ones
> • Code movement
> ● Moving a block of code outside a loop
> • Dead code elimination
> ● Eliminated code statements that are either never executed or unreachable

Compiler Design module | PDF page 18 | lines 32-41

> • Constant folding
> ● Eg.
> ● R2 = 3 * 2
> • Constant Propagation
> ● Eg.
> ● r1 = 3
> ● r2 = r1 * 2
> • Copy Propagation
> ● Eg.
> ● r2 = r1

Question sources

MOE Computer Science Exit Exam · Question 8-1

National Computer Science Exit Exam · Question 5

National Computer Science Exit Exam · Question 6

National Computer Science Exit Exam

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 263

Question 82 | SM0041

Which one of the following phases is done BEFORE the semantic analysis phase of the compiling process?

A. Syntax analysis

B. Intermediate code generation

C. Code generation

D. Code optimization

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Syntax analysis

PDF pp5-7 orders syntax analysis before semantic analysis, followed by intermediate generation, optimization, and code generation.

Why each choice is right or wrong

A - Correct: Syntax analysis appears on p5 before semantics on p6.

B - Incorrect: Intermediate generation follows semantic analysis on p6.

C - Incorrect: Code generation is phase 6 on p7.

D - Incorrect: Optimization is phase 5 on p7, after semantics.

Primary note proof

Study-note section COMP03

> The conventional sequence is lexical analysis → syntax analysis → semantic analysis → intermediate-code generation → optimization → target-code generation. Scanning turns characters into tokens; parsing builds structure; semantics checks declarations and types; synthesis produces implementation code. A phase is a logical activity; a pass is a traversal that can combine activities.

Study-note section COMP03

> The front end handles source-language rules; a middle end analyzes and transforms IR; the backend handles target details. Symbol-table management and error reporting support multiple phases rather than forming one mandatory seventh final phase. Optimization can occur at several levels, including after instruction selection. "Final phase" means code generation in the standard six-phase teaching pipeline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 6 | lines 11-12

4. Intermediate Code Generation
Easy to produce and easy to translate to machine code

Compiler Design module | PDF page 7 | lines 2-6

5. Code Optimization
Changes the IC by removing such inefficiencies
Improve the code
a. Improvement may be time, space, or power consumption.
It changes the structure of programs,

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Question sources

MOE Computer Science Exit Exam · Question 12-3

National Computer Science Exit Exam · Question 87

National Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 264

Question 83 | SM0052

Which phase of the compiling process is considered the FINAL phase?

A. Intermediate code generation

B. Semantic analysis

C. Code optimization

D. Code generation

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Code generation in the conventional pipeline.

COMP03 calls code generation the final phase of the standard six-phase teaching pipeline. Intermediate generation, semantic analysis and optimization precede it there. Some implementations perform later target optimization, but that does not change the conventional phase answer requested here; the note explicitly records that distinction.

Why each choice is right or wrong

A - Incorrect: Intermediate-code generation precedes optimization and target-code generation in the conventional sequence.

B - Incorrect: Semantic analysis checks names and types before intermediate and target synthesis.

C - Incorrect: Optimization precedes final code generation in the simplified six-phase teaching pipeline.

D - Correct: COMP03 explicitly names code generation as final under this standard pipeline convention.

Primary note proof

Study-note section COMP03

> The conventional sequence is lexical analysis → syntax analysis → semantic analysis → intermediate-code generation → optimization → target-code generation. Scanning turns characters into tokens; parsing builds structure; semantics checks declarations and types; synthesis produces implementation code. A phase is a logical activity; a pass is a traversal that can combine activities.

Study-note section COMP03

> The front end handles source-language rules; a middle end analyzes and transforms IR; the backend handles target details. Symbol-table management and error reporting support multiple phases rather than forming one mandatory seventh final phase. Optimization can occur at

several levels, including after instruction selection. “Final phase” means code generation in the standard six-phase teaching pipeline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Compiler Design module | PDF page 7 | lines 2-6

5. Code Optimization
Changes the IC by removing such inefficiencies
Improve the code
a. Improvement may be time, space, or power consumption.
It changes the structure of programs,

Question sources

MOE Computer Science Exit Exam · Question 15-3

MOE Computer Science Exit Exam

Computer Science Exit Exam practice collection · Question 265

Question 84 | SM0093

Which one of the following is used for the purpose of syntax analysis?

A. Loader

B. Linker

C. Parser

D. Scanner

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: Parser

PDF p10 defines parsing as constructing a parse tree, and p5 labels syntax analysis Parsing. P3 assigns distinct object-code roles to loader and linker.

Why each choice is right or wrong

A - Incorrect: P3 loads an executable into memory.

B - Incorrect: P3 merges object files.

C - Correct: Parsing is syntax analysis (p5).

D - Incorrect: Scanning produces tokens (p8), prior to parsing.

Primary note proof

Study-note section COMP02

A preprocessor handles directives such as file inclusion and macros. An assembler translates assembly to object code. A linker combines object files and libraries, resolves external symbols and performs relocation. A loader maps an executable into memory and starts it.

Study-note section COMP03

Scanning turns characters into tokens; parsing builds structure; semantics checks declarations and types; synthesis produces implementation code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 10 | lines 11-17

Parsing
• Process of constructing parse tree for a sentence generated by a given grammar.
• 2 types of parsers
● Top down parsing (predictive parsers)
• LL(1)
● Bottom up parsing (SR parsers)
• LR(1)

Compiler Design module | PDF page 3 | lines 9-16

• Assembler
● assembly language into machine code
● output of an assembler is called an object file
• Linker
● links and merges various object files to make an executable file.
● determine the memory location where these codes will be loaded
• Loader
● loading executable files into memory and execute them.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Question sources

MOE Computer Science Exit Exam · Question 26-3

Computer Science Exit Exam practice collection · Question 266

Question 85 | SM0216

Which output does syntax analysis produce when it retains every grammar production and terminal occurrence, including punctuation, without semantic attribute annotation?

A. Concrete syntax tree (parse tree)

B. Annotated parse tree

C. Abstract syntax tree

D. Pattern

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer A: Concrete syntax tree (parse tree)

A concrete syntax or parse tree records the grammar productions used, with nonterminal interiors and terminal leaves. Because the specified representation retains grammar scaffolding and punctuation, it is not the abstract syntax tree that omits those details. Semantic attribute values would make an annotated tree, which this question excludes. A lexical pattern is a specification for tokens rather than the parsed program representation.

Why each choice is right or wrong

A - Correct: A full syntax/parse tree records the retained grammar detail.

B - Incorrect: Semantic attribute annotation is expressly excluded.

C - Incorrect: An AST omits the scaffolding and punctuation retained here.

D - Incorrect: A pattern specifies lexical membership rather than parsed program structure.

Primary note proof

Study-note section COMP24

A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions. An annotated parse tree displays computed attribute values; it is not itself an inherited or synthesized attribute.

Study-note section COMP09

A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input. An AST omits grammar scaffolding and punctuation; expression operands are leaves and operators internal nodes. The parser can build either representation, depending on implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 9 | lines 22-27

Derivation Trees
• Derivations can be displayed as trees
• Internal nodes of the tree are all non-terminals
• Leaves are all terminals
• The yield of a derivation tree is the list of the labels of all the leaves read from left to right.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 5-3

Computer Science Exit Exam practice collection · Question 270

Question 86 | SM0219

For a fixed suitable grammar, which listed approaches can be efficient top-down parsers using prediction without backtracking? Select all that apply.

A. Recursive-descent parser

B. Shift-reduce parser

C. Non-recursive predictive parser

D. LR parser

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer A, C: Recursive-descent parser; Non-recursive predictive parser

Recursive descent can use lookahead to predict productions without backtracking. Nonrecursive predictive parsing implements the same top-down strategy with an explicit stack and table. Both can run linearly for a fixed suitable grammar. Shift-reduce and LR instead work bottom-up, reducing input toward the start symbol, so they do not meet the specified parsing direction even if they are efficient.

Why each choice is right or wrong

A - Correct: Predictive recursive descent is a supported efficient top-down implementation.

B - Incorrect: Shift-reduce is bottom-up.

C - Correct: Nonrecursive prediction is a supported efficient top-down implementation.

D - Incorrect: LR is bottom-up.

Primary note proof

Study-note section COMP11

> Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP11

> A nonrecursive predictive parser stores grammar symbols on an explicit stack and consults a table. It is another implementation of predictive parsing, not a wholly different language class. Table-driven and recursive predictive implementations can both be linear for a fixed suitable grammar. Efficiency cannot be ranked from their names alone.

Study-note section COMP15

> Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 18-23

> Top Down Parsing
> • The parse tree is created top to bottom
> • Starts from the start symbol and transform it to the input
> Bottom Up Parsing
> • Starts with the input symbols and tries to construct the parse tree up to the start symbol.
> • One way of reducing a sentence is to follow the right most derivation in reverse

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science Exit Exam · Sene 2016 · Question 6-3

Question 87 | SM0224

Which abstract data structure naturally represents hierarchical source-program constructs with parent/child relationships, as in a parse tree or AST?

A. 2D-array

B. Stack

C. Tree

D. Queue

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: Tree

A tree represents the hierarchy of program constructs through parent/child relationships. Parse trees preserve grammar detail, while ASTs retain the essential expression and statement structure. Arrays may implement node storage, and stacks or queues may support traversal, but those container roles do not make them the requested hierarchical model. The structure being represented, rather than the storage allocation technique, determines this answer.

Why each choice is right or wrong

A - Incorrect: A 2D array is not the requested abstract hierarchical model.

B - Incorrect: A stack supports processing order rather than the full parent/child model.

C - Correct: A tree directly models hierarchical constructs.

D - Incorrect: A queue supports work ordering rather than the hierarchy itself.

Primary note proof

Study-note section COMP41

> Grouping characters into token classes is lexical analysis; grouping tokens into expressions and statements is syntax analysis.

Study-note section COMP41

> Trees store hierarchical constructs, while a symbol table stores declaration metadata.

Study-note section COMP09

> A context-free grammar G=(N,T,P,S) has nonterminals N, terminals T, productions P and start S. A production A→α rewrites one nonterminal. A sentential form is any string of terminals/nonterminals derivable from S; a sentence contains only terminals. A leftmost derivation rewrites the leftmost remaining nonterminal; a rightmost one rewrites the rightmost.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input. An AST omits grammar scaffolding and punctuation; expression operands are leaves and operators internal nodes. The parser can build either representation, depending on implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

> 3. Semantic Analysis
> • Certain checks are performed to make sure that the components of the program fit together meaningfully.
> • Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
> - Type checking of various programming language constructs is one of the most important tasks.
> • Stores type information in the symbol table or the syntax tree.
> - Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 13 | lines 2-14

> ● Two fields containing pointers to left and right
> • mkleaf(id, entry)
> ● Creates an identifier node with label id &
> ● A field containing entry, ptr to symbol table entry for the identifier
> • mkleaf(num, val)
> ● Create a number node with label num &
> ● A field containing val, the value for the number
> Syntax tree for expression a-4+c

• P1=mkleaf(id,entrya);
• P2=mkleaf(num, 4);
• P3=mknode('-',p1,p2);
• P4=mkleaf(id,entryc);
• P5=mknode('+',p3,p4);

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float
id.width=4
offset = offset +id.width { offset = 6 }

Question sources

Computer Science Exit Exam · Sene 2016 · Question 8-2

Computer Science Exit Exam practice collection · Question 142

Question 88 | SM0238

In which phase of the compilation process does the source program get broken into its constituent parts?

A. Semantic Analysis

B. Syntax Analysis

C. Synthesis

D. Lexical Analysis

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Lexical Analysis

PDF p5 scans characters into lexemes represented as tokens. These constituent lexical units precede hierarchical grouping during parsing.

Why each choice is right or wrong

A - Incorrect: P6 checks meaningful compatibility after lexical decomposition.

B - Incorrect: Parsing groups already recognized tokens (p5).

C - Incorrect: P3 calls synthesis machine dependent; it is not the scanning stage.

D - Correct: P5 decomposes the character stream into tokens.

Primary note proof

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

1. Lexical / Linear Analysis (scanning)
- Scans the source code as a stream of characters
- Represent lexemes in the form of tokens as:
<token-name, attribute-value>
- Token
• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 12-3

Computer Science Exit Exam practice collection · Question 269

Question 89 | SM0265

Which described check requires runtime values rather than being solely a static check? In D, the index depends on input and no static proof of its bounds is available.

A. Type checking for compatibility

B. Uniqueness checks like redefined variables

C. Checking that break is inside an allowed loop or switch

D. Checking an input-dependent array index against runtime bounds, with no static proof available

Hint

Ask whether the required information is a declaration's identity and attributes, program structure or the ordered source tokens.

Answer D: Checking an input-dependent array index against runtime bounds, with no static proof available

Type compatibility, duplicate declarations and valid placement of break can be checked from the source program and its declarations. The specified array index depends on runtime input, and no static bounds proof is available, so its validity requires a runtime check. This does not claim that every memory access fault is inherently undetectable statically; the explicit value-dependent case supplies the distinction.

Why each choice is right or wrong

A - Incorrect: Type compatibility is a static semantic check.

B - Incorrect: Duplicate declarations can be checked statically within the relevant scope.

C - Incorrect: Placement in an allowed loop or switch is a static check.

D - Correct: The stipulated input-dependent bounds need runtime validation.

Primary note proof

Study-note section COMP21

> Static checks occur before execution: type compatibility, duplicate declarations within a restricted scope, valid names, argument counts, return compatibility and control-flow placement. A break is valid only in constructs allowed by the language; in C/C++ it can exit a loop or switch. A compiler may annotate an AST or create typed IR; there is no mandatory uniquely named output tree.

Study-note section COMP21

> Dynamic checks depend on runtime values, such as some bounds and null checks. Static analysis can prove some accesses safe or faulty; "all memory errors cannot be statically checked" is too

strong. Unreachable code is not universally a semantic error: some languages reject it in specified cases, others merely warn or optimize it away.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 14 | lines 16-23

Static Checking
• The compiler must check the semantic conventions of the source language
• Static Checking: ensures that certain kind of errors are detected and reported
• Example
• Type Checks: incompatible operands
• Flow Control Check
• Uniqueness Check
• Name Related Check

Compiler Design module | PDF page 14 | lines 2-15

• Type Errors:
- Improper or inconsistent operations during program execution
• Type-safety:
- Absence of type errors

Type Checking
• Semantic checks to enforce the type safety of the program
• Semantic Checks
- Static - done during compilation
- Dynamic - done during run-time
• Examples
- Unary and binary operators
- Number and type of arguments
- Return statement with return type
- Compatible assignment

Question sources

Computer Science Exit Exam · Sene 2016 · Question 21-2

Computer Science Exit Exam practice collection · Question 271

Question 90 | SM0285

Which tool is used to collect codes separately compiled or assembled in different object files into a file that is directly executable?

A. Linker

B. Loader

C. Assembler

D. Preprocessor

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Linker

A linker combines separately compiled object files and required libraries, resolves their external symbol references and relocates addresses to produce the executable form. An assembler translates assembly into object code before this combination. A loader places the resulting executable in memory and starts it. A preprocessor handles source directives before compilation. These tool roles distinguish object-file combination from translation, source expansion and program loading.

Why each choice is right or wrong

A - Correct: The linker combines object files, resolves symbols and performs relocation.

B - Incorrect: The loader maps an already produced executable into memory.

C - Incorrect: An assembler translates assembly source into object code.

D - Incorrect: A preprocessor handles source directives rather than combining object files.

Primary note proof

Study-note section COMP02

> A preprocessor handles directives such as file inclusion and macros. An assembler translates assembly to object code. A linker combines object files and libraries, resolves external symbols and performs relocation. A loader maps an executable into memory and starts it. Compilation alone does not necessarily produce a ready executable.

Study-note section COMP02

> A cross-compiler runs on a host platform and generates code for a different target. Multiple targets require appropriate backends, not a special definition of cross-compilation. A retargetable compiler supports changing target descriptions. Incremental compilation recompiles affected fragments or dependent units after edits; it is more than literally replacing changed lines. One-pass versus multipass counts traversals, not supported targets.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 3 | lines 9-16

> • Assembler
> - assembly language into machine code
> - output of an assembler is called an object file
> • Linker
> - links and merges various object files to make an executable file.
> - determine the memory location where these codes will be loaded
> • Loader
> - loading executable files into memory and execute them.

Question sources

Computer Science Exit Exam · Sene 2016 · Question 28-1

Computer Science Exit Exam practice collection · Question 272

Question 91 | SS059

Among phases of compiling process, in which phase a sequence of characters are converted into a tokens?

A. Code optimization

B. Syntax analysis

C. Semantic analysis

D. Lexical analysis

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Lexical analysis

PDF pp5,8 gives lexical analysis a character-stream input and a sequence-of-tokens output.

Why each choice is right or wrong

A - Incorrect: Optimization changes intermediate code (p7).

B - Incorrect: Syntax analysis groups tokens already recognized (p5).

C - Incorrect: Semantic analysis checks meaningful fit (p6).

D - Correct: The exact input-output description is on p8.

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

1. Lexical / Linear Analysis (scanning)
- Scans the source code as a stream of characters
- Represent lexemes in the form of tokens as:

<token-name, attribute-value>
- Token

• smallest meaningful element that a compiler understands.
- Eg.

• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Question sources

National Computer Science Exit Exam · Question 61

National Computer Science Exit Exam

Question 92 | AM0779

Which phase of the compiler is responsible for recognizing tokens and building a symbol table?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Ask whether the required information is a declaration’s identity and attributes, program structure or the ordered source tokens.

Answer A: Lexical Analysis

Token recognition uniquely identifies lexical analysis on p5. However, p6 and p17 assign type/name entries to semantic and declaration processing, so building the symbol table must mean contributing lexical identifier entries, not constructing every attribute exclusively.

Why each choice is right or wrong

A - Correct: P5 explicitly recognizes tokens; A is supported for that decisive responsibility.

B - Incorrect: Parsing groups tokens rather than recognizing the character stream (p5).

C - Incorrect: P6 stores types in the symbol table but does not perform lexical token recognition.

D - Incorrect: P7 converts intermediate code into machine code.

Primary note proof

Study-note section COMP19

Scanning may intern a spelling first; semantic analysis determines the declaration and type. Implementations differ, so first interaction is conventional rather than universal.

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

1. Lexical / Linear Analysis (scanning)
- Scans the source code as a stream of characters
- Represent lexemes in the form of tokens as:
<token-name, attribute-value>
- Token
• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address

```
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float
id.width=4
offset = offset +id.width { offset = 6 }
```

Question sources

Computer Science MCQ practice collection · Question 779

Question 93 | AM0780

What is the primary ordered output of lexical analysis passed to the parser?

A. Abstract Syntax Tree (AST)

B. Ordered token stream

C. Intermediate Code

D. Assembly Code

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Ordered token stream

Lexical analysis groups characters into lexemes and emits an ordered token stream, including token categories and relevant attributes. The parser consumes that stream to build grammatical structure. A symbol table may be updated as an auxiliary service, but it is not the primary ordered output. ASTs, intermediate code and assembly belong to later processing stages in the conventional pipeline.

Why each choice is right or wrong

A - Incorrect: AST construction follows scanning during parsing or syntax-directed work.

B - Correct: The ordered token stream is the parser's input.

C - Incorrect: Intermediate code follows syntactic and semantic processing.

D - Incorrect: Assembly is a possible later target output.

Primary note proof

Study-note section COMP41

Grouping characters into token classes is lexical analysis; grouping tokens into expressions and statements is syntax analysis.

Study-note section COMP41

Trees store hierarchical constructs, while a symbol table stores declaration metadata.

Study-note section COMP04

A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Question sources

Computer Science MCQ practice collection · Question 780

Computer Science MCQ practice collection · Question 790

Question 94 | AM0781

Which phase of the compiler is responsible for checking the correctness of the program's structure?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Syntax Analysis

PDF p5 groups tokens by grammar and catches most syntactic errors. Program structure here means syntactic structure, so B applies.

Why each choice is right or wrong

A - Incorrect: Recognizes lexical units rather than their grammatical arrangement (p5).

B - Correct: Checks grammar-governed hierarchy and syntactic errors (p5).

C - Incorrect: Checks meaning and type mismatch (p6).

D - Incorrect: Translates intermediate code into machine code (p7).

Primary note proof

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

> 2. Syntax / Hierarchical Analysis - Parsing
> • Tokens are grouped hierarchically into nested collections with collective meaning.
> • The result is generally a parse tree.
> • expressions, statements, declarations etc... are identified by using the results of lexical analysis.
> • Most syntactic errors in the source program are caught in this phase.
> • Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 6 | lines 2-10

> 3. Semantic Analysis
> • Certain checks are performed to make sure that the components of the program fit together meaningfully.
> • Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
> - Type checking of various programming language constructs is one of the most

important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Computer Science MCQ practice collection · Question 781

Question 95 | AM0782

Which component of the compiler generates an Abstract Syntax Tree (AST)?

A. Lexical Analyzer

B. Parser

C. Semantic Analyzer

D. Code Generator

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Parser

PDF p5 gives parsing the tree-building role. P12 says syntax-directed actions execute according to parsing, and pp12-13 give operator/operand syntax-tree constructors. Together these support parser as the component building the abstract structural tree.

Why each choice is right or wrong

A - Incorrect: Lexical analysis emits tokens (p8).

B - Correct: Parsing builds a tree (p5), with syntax-directed tree constructors on pp12-13.

C - Incorrect: Semantic analysis may annotate the existing tree (p6); it is not the phase assigned general syntactic tree construction.

D - Incorrect: Code generation translates intermediate code (p7).

Primary note proof

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 12 | lines 4-16

Syntax Directed Translation
• Attaching actions to the grammar rules(productions).
• Actions are executed during the compilation
- Not during the generation of the compiler
• Actions are executed according to the parsing mechanism.
Syntax Directed Definitions
• Is a generalization of a context free grammar
• Is a CFG with attributes and rules
• Attributes are associated with grammar symbols and rules with productions
• Attributes may be:
- Numbers
- Types
- Strings etc

Compiler Design module | PDF page 12 | lines 26-28

Functions for Syntax Tree Nodes
• mknode (op, left, right)
- Creates an operator node with label op &

Compiler Design module | PDF page 13 | lines 2-14

- Two fields containing pointers to left and right
• mkleaf(id, entry)
- Creates an identifier node with label id &
- A field containing entry, ptr to symbol table entry for the identifier
• mkleaf(num, val)
- Create a number node with label num &
- A field containing val, the value for the number
Syntax tree for expression a-4+c
• P1=mkleaf(id,entrya);
• P2=mkleaf(num, 4);
• P3=mknode('-',p1,p2);
• P4=mkleaf(id,entryc);
• P5=mknode('+',p3,p4);

Question sources

Computer Science MCQ practice collection · Question 782

Computer Science MCQ practice collection · Question 792

Question 96 | AM0783

What does the semantic analysis phase of the compiler focus on?

A. Identifying syntax errors

B. Analyzing the meaning of program statements

C. Generating intermediate code

D. Translating source code to machine code

Hint

Ask whether the required information is a declaration's identity and attributes, program structure or the ordered source tokens.

Answer B: Analyzing the meaning of program statements

PDF p6 defines semantic analysis through meaningful fit and type-mismatch checks, directly supporting B.

Why each choice is right or wrong

A - Incorrect: Most syntactic errors are assigned to parsing on p5.

B - Correct: Meaningful compatibility is the explicit semantic task (p6).

C - Incorrect: Intermediate generation is separately headed on p6.

D - Incorrect: Machine-code translation is code generation (p7).

Primary note proof

Study-note section COMP21

> Static checks occur before execution: type compatibility, duplicate declarations within a restricted scope, valid names, argument counts, return compatibility and control-flow placement. A break is valid only in constructs allowed by the language; in C/C++ it can exit a loop or switch. A compiler may annotate an AST or create typed IR; there is no mandatory uniquely named output tree.

Study-note section COMP21

> Dynamic checks depend on runtime values, such as some bounds and null checks. Static analysis can prove some accesses safe or faulty; "all memory errors cannot be statically checked" is too strong. Unreachable code is not universally a semantic error: some languages reject it in specified cases, others merely warn or optimize it away.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 6 | lines 11-12

4. Intermediate Code Generation
Easy to produce and easy to translate to machine code

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Question sources

Computer Science MCQ practice collection · Question 783

Question 97 | AM0784

Which phase of the compiler performs type checking and enforces semantic rules?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: Semantic Analysis

PDF p6 explicitly names type checking as one of the most important semantic-analysis tasks.

Why each choice is right or wrong

A - Incorrect: Scanning recognizes tokens (p5).

B - Incorrect: Parsing checks grammatical hierarchy (p5).

C - Correct: Type checking is explicitly here (p6).

D - Incorrect: Code generation converts intermediate code (p7).

Primary note proof

Study-note section COMP22

> A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

> Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

> 3. Semantic Analysis
> • Certain checks are performed to make sure that the components of the program fit together meaningfully.
> • Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
> - Type checking of various programming language constructs is one of the most important tasks.
> • Stores type information in the symbol table or the syntax tree.
> - Types of variables, function parameters, array dimensions, etc.

Question sources

Computer Science MCQ practice collection · Question 784

Question 98 | AM0785

Which phase of the compiler translates the intermediate representation into the target machine code?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Code Generation

PDF p7 states that code generation converts intermediate code to machine code.

Why each choice is right or wrong

A - Incorrect: Scanning works on source characters (p5).

B - Incorrect: Parsing groups tokens (p5).

C - Incorrect: Semantic analysis checks types and meaning (p6).

D - Correct: Matches the explicit conversion on p7.

Primary note proof

Study-note section COMP34

> Code generation lowers checked IR to a target form. Instruction selection chooses target operations; register allocation chooses where values reside; scheduling orders instructions while respecting dependencies; storage layout assigns offsets and follows ABI requirements. Register assignment names actual physical registers after allocation decisions.

Study-note section COMP34

> For t=a+b, a simple target could emit LOAD R1,a; LOAD R2,b; ADD R3,R1,R2; STORE t,R3. Other targets may combine loads with arithmetic or keep t in a register. Machine idioms, register availability and instruction order affect execution time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 7-11

> 6. Code Generation
> Converts intermediate code to machine code.
> Must handle all aspects of machine architecture
> Storage allocation decisions are made
> a. Register allocation and assignment

Question sources

Computer Science MCQ practice collection · Question 785

Computer Science MCQ practice collection · Question 794

Question 99 | AM0786

What is the primary goal of the code optimization phase in compiler design?

A. To reduce the size of the source code

B. To eliminate syntax errors

C. To improve the performance of the generated code

D. To generate human-readable code

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer C: To improve the performance of the generated code

PDF p7 names improvement in time, space, or power; p17 adds faster execution, efficient memory, and preserving meaning. Thus improving generated-code performance is the best statement.

Why each choice is right or wrong

A - Incorrect: Source text length is not the optimization goal.

B - Incorrect: Syntax errors are caught by parsing (p5); optimization assumes meaningful code to improve.

C - Correct: Matches the performance goal on p17.

D - Incorrect: Human readability is not the goal listed on pp7,17.

Primary note proof

Study-note section COMP30

> Constant folding evaluates constant expressions during compilation: 3*4→12. Constant propagation substitutes known values: x=3; y=x+2 becomes y=3+2 before folding. Copy propagation substitutes a copied name while its source remains unchanged. Dead-code elimination removes unreachable code or computations whose values are unused and whose evaluation has no necessary side effects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 2-6

5. Code Optimization
Changes the IC by removing such inefficiencies
Improve the code
a. Improvement may be time, space, or power consumption.
It changes the structure of programs,

Compiler Design module | PDF page 17 | lines 16-22

Chapter 8
Introduction to Code Optimization
Goals of Code Optimization
• Remove redundant code without changing the meaning of program
• Executes faster
• Efficient memory usage
• Better performance

Question sources

Computer Science MCQ practice collection · Question 786

Question 100 | AM0787

For E→E+E|E*E|id, which technique resolves the ambiguity by requiring * to bind more tightly than + and both operators to associate left?

A. Parsing with explicit operator precedence and left-associativity rules

B. Lexical analysis

C. Semantic analysis

D. Leftmost derivation

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Parsing with explicit operator precedence and left-associativity rules

The grammar permits multiple groupings of expressions such as id+id*id. An explicit precedence and associativity policy selects multiplication before addition and left association within each level. A parser may enforce this through declarations or a grammar rewrite. Lexical analysis and generic semantic checking do not themselves establish that policy, and taking a leftmost derivation does not remove the competing parse trees.

Why each choice is right or wrong

A - Correct: The stated policy selects the intended grouping.

B - Incorrect: Token recognition does not determine operator grouping.

C - Incorrect: Generic semantic checking supplies no stated grouping policy.

D - Incorrect: An ambiguous grammar can have multiple leftmost derivations.

Primary note proof

Study-note section COMP10

A grammar is ambiguous if at least one terminal string has two different parse trees, equivalently two leftmost derivations. Merely changing derivation order on the same tree does not establish ambiguity. E→E+E|E*E|id allows different trees for id+id*id.

Study-note section COMP10

Use E→E+T|T, T→T*F|F, F→(E)|id to give * higher precedence and both operators left associativity. For subtraction, a−b−c then groups (a−b)−c. The dangling-else grammar permits else to attach to either nested if; a nearest-unmatched-if rule or grammar rewrite resolves it. Taking a leftmost derivation by itself does not resolve ambiguous interpretation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 6-16

Grammars
• Every programming language has grammar rules
• Parsers or syntax analyzers are generated for a particular grammar
• CFG are used for syntax specification of programming languages
Context Free Grammar (CFG)
• Is denoted as G = (N, T , P, S)
• N : finite set of non-terminals
• T : finite set of terminals
• S ∈ N: The start symbol
• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Compiler Design module | PDF page 10 | lines 11-17

Parsing
• Process of constructing parse tree for a sentence generated by a given grammar.
• 2 types of parsers
● Top down parsing (predictive parsers)
• LL(1)
● Bottom up parsing (SR parsers)
• LR(1)

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number

of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science MCQ practice collection · Question 787

Question 101 | AM0788

What is the purpose of semantic rules in a programming language?

A. To define the syntax of the language

B. To specify the order of execution of statements

C. To enforce the correct use of language constructs

D. To optimize the generated code

Hint

Ask whether the required information is a declaration’s identity and attributes, program structure or the ordered source tokens.

Answer C: To enforce the correct use of language constructs

PDF p6 checks that program components fit together meaningfully and p14 requires semantic conventions to hold. C captures those constraints on use of language constructs.

Why each choice is right or wrong

A - Incorrect: Grammatical structure is assigned to syntax analysis (p5).

B - Incorrect: Execution order alone does not capture the type/compatibility constraints on pp6,14.

C - Correct: Correct semantic use follows the cited compatibility and convention checks.

D - Incorrect: Optimization is separately concerned with efficiency (p7).

Primary note proof

Study-note section COMP21

Static checks occur before execution: type compatibility, duplicate declarations within a restricted scope, valid names, argument counts, return compatibility and control-flow placement. A break is valid only in constructs allowed by the language; in C/C++ it can exit a loop

or switch. A compiler may annotate an AST or create typed IR; there is no mandatory uniquely named output tree.

Study-note section COMP21

Dynamic checks depend on runtime values, such as some bounds and null checks. Static analysis can prove some accesses safe or faulty; “all memory errors cannot be statically checked” is too strong. Unreachable code is not universally a semantic error: some languages reject it in specified cases, others merely warn or optimize it away.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 14 | lines 16-23

Static Checking
• The compiler must check the semantic conventions of the source language
• Static Checking: ensures that certain kind of errors are detected and reported
• Example
• Type Checks: incompatible operands
• Flow Control Check
• Uniqueness Check
• Name Related Check

Question sources

Computer Science MCQ practice collection · Question 788

Question 102 | AM0789

Which of the following is responsible for recognizing keywords and identifiers in a programming language?

A. Lexical Analyzer

B. Parser

C. Semantic Analyzer

D. Code Generator

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Lexical Analyzer

PDF p5 lists identifiers and keywords among lexical tokens, and p8 explains their token classes.

Why each choice is right or wrong

A - Correct: Recognizes the token classes explicitly listed on pp5,8.

B - Incorrect: Parser consumes tokens already recognized (p9).

C - Incorrect: Semantic analysis checks type and meaning (p6).

D - Incorrect: Code generation produces machine code (p7).

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

> 1. Lexical / Linear Analysis (scanning)
> - Scans the source code as a stream of characters
> - Represent lexemes in the form of tokens as:
> <token-name, attribute-value>
> - Token
> • smallest meaningful element that a compiler understands.
> - Eg.
> • Identifiers, Keywords, Literals, Operators and Special symbols.
> - Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 8 | lines 19-24

> • Classes of Tokens
> - Identifiers: names chosen by the programmer
> - Keywords: names already in the programming language

- Separators: punctuation characters
- Operators: symbols that operate on arguments and produce results
- Literals: numeric, textual literals

Question sources

Computer Science MCQ practice collection · Question 789

Question 103 | AM0791

Which phase of the compiler is responsible for checking the syntax of the program?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Syntax Analysis

PDF p5 names syntax analysis Parsing and assigns it grammatical hierarchy and most syntactic-error detection.

Why each choice is right or wrong

A - Incorrect: Scanning identifies lexemes, not grammar structure (p5).

B - Correct: Directly matches the syntax-checking phase on p5.

C - Incorrect: P6 concerns semantic errors such as type mismatch.

D - Incorrect: P7 concerns machine-code generation.

Primary note proof

Study-note section COMP04

Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

> 2. Syntax / Hierarchical Analysis - Parsing
> • Tokens are grouped hierarchically into nested collections with collective meaning.
> • The result is generally a parse tree.
> • expressions, statements, declarations etc... are identified by using the results of lexical analysis.
> • Most syntactic errors in the source program are caught in this phase.
> • Syntactic rules of the source language are given via a Grammar.

Question sources

Computer Science MCQ practice collection · Question 791

Question 104 | AM0793

What is the main goal of the semantic analysis phase in compiler design?

A. To check the syntax of the program

B. To generate intermediate code

C. To perform type checking and enforce semantic rules

D. To translate the source code to machine code

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: To perform type checking and enforce semantic rules

PDF p6 checks meaningful fit, detects semantic errors, and explicitly emphasizes type checking.

Why each choice is right or wrong

A - Incorrect: Syntactic structure is checked in parsing (p5).

B - Incorrect: Intermediate generation is a separate phase (p6).

C - Correct: Matches both semantic compatibility and type checking on p6.

D - Incorrect: Machine-code conversion belongs to code generation (p7).

Primary note proof

Study-note section COMP22

> A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Computer Science MCQ practice collection · Question 793

Question 105 | AM0795

What is the purpose of an Abstract Syntax Tree (AST)?

A. To store the program's symbol table

B. To generate assembly code

C. To visualize the program's control flow

D. To represent the hierarchical structure of the program

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: To represent the hierarchical structure of the program

PDF pp12-13 represents a-4+c through identifier/number leaves and operator nodes linked by child pointers. This encodes the expression hierarchy, supporting D.

Why each choice is right or wrong

A - Incorrect: P13 leaves may point to a symbol-table entry, but the whole tree is not the symbol table.

B - Incorrect: P7 assigns machine-code production to code generation; that is not the tree definition.

C - Incorrect: The illustrated tree encodes expression operators and operands, not control-flow paths.

D - Correct: Operator-child nesting on pp12-13 represents program syntax hierarchy.

Primary note proof

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

> 2. Syntax / Hierarchical Analysis - Parsing
> • Tokens are grouped hierarchically into nested collections with collective meaning.
> • The result is generally a parse tree.
> • expressions, statements, declarations etc... are identified by using the results of lexical analysis.
> • Most syntactic errors in the source program are caught in this phase.
> • Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 12 | lines 26-28

> Functions for Syntax Tree Nodes
> • mknode (op, left, right)
> ● Creates an operator node with label op &

Compiler Design module | PDF page 13 | lines 2-14

> ● Two fields containing pointers to left and right
> • mkleaf(id, entry)
> ● Creates an identifier node with label id &
> ● A field containing entry, ptr to symbol table entry for the identifier
> • mkleaf(num, val)
> ● Create a number node with label num &
> ● A field containing val, the value for the number
> Syntax tree for expression a-4+c
> • P1=mkleaf(id,entrya);
> • P2=mkleaf(num, 4);

- P3=mknode('-',p1,p2);
- P4=mkleaf(id,entryc);
- P5=mknode('+',p3,p4);

Question sources

Computer Science MCQ practice collection · Question 795

Question 106 | AM0796

What does the symbol table in a compiler store?

A. Intermediate code

B. Machine code

C. Source code

D. Information about identifiers and their attributes

Hint

Ask whether the required information is a declaration's identity and attributes, program structure or the ordered source tokens.

Answer D: Information about identifiers and their attributes

PDF p17 puts names and types in the symbol table; p6 adds variables, function parameters, and array dimensions. P13 points identifier leaves to table entries. This is identifier information and attributes.

Why each choice is right or wrong

A - Incorrect: Intermediate code has separate statement representations (pp15-16).

B - Incorrect: Machine code is output of code generation (p7).

C - Incorrect: The cited table entries are names/types, not the full source text.

D - Correct: Names and type attributes are explicitly stored (pp6,17).

Primary note proof

Study-note section COMP19

A symbol table maps names or symbol identities to attributes: type, scope, width, storage offset, parameter signature, constant information or linkage. A hash table gives expected O(1) insertion/lookup with suitable hashing; a stack of scope tables supports lexical nesting. Search the nearest scope first, insert declarations in the current scope, and remove visibility when leaving it.

Study-note section COMP19

Scanning may intern a spelling first; semantic analysis determines the declaration and type. Implementations differ, so first interaction is conventional rather than universal. Not every

punctuation token needs a symbol-table entry; literals may use a separate literal table. A hash table is an implementation, not the information’s semantic role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float
id.width=4
offset = offset +id.width { offset = 6 }

Compiler Design module | PDF page 13 | lines 2-14

- Two fields containing pointers to left and right
• mkleaf(id, entry)
- Creates an identifier node with label id &
- A field containing entry, ptr to symbol table entry for the identifier
• mkleaf(num, val)
- Create a number node with label num &
- A field containing val, the value for the number
Syntax tree for expression a-4+c
• P1=mkleaf(id,entrya);
• P2=mkleaf(num, 4);
• P3=mknode(‘-’,p1,p2);
• P4=mkleaf(id,entryc);
• P5=mknode(‘+’,p3,p4);

Question sources

Computer Science MCQ practice collection · Question 796

Question 107 | AM0797

Which phase of the compiler performs type checking?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: Semantic Analysis

PDF p6 explicitly places type checking in semantic analysis.

Why each choice is right or wrong

A - Incorrect: Scanning turns characters into tokens (p5).

B - Incorrect: Parsing groups tokens and checks syntax (p5).

C - Correct: P6 explicitly names this task.

D - Incorrect: Generation converts intermediate to machine code (p7).

Primary note proof

Study-note section COMP22

> A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

> Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Computer Science MCQ practice collection · Question 797

Question 108 | AM0798

What is the purpose of intermediate code in compiler design?

A. To optimize the program's execution

B. To generate assembly code

C. To represent the program's semantics in a platform-independent form

D. To visualize the program's control flow

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer C: Represent program meaning in a reusable, substantially platform-independent intermediate form.

COMP26 identifies separation of source analysis and target implementation as IR's architectural purpose, allowing reuse across targets. C expresses that purpose best. Optimization and assembly generation are operations enabled by the representation, while a control-flow graph is one possible representation aspect. Platform independence is a design benefit here, not a claim that every IR has no target details.

Why each choice is right or wrong

A - Incorrect: Optimization is enabled by IR, but it is not the general representational purpose distinguishing this choice.

B - Incorrect: Assembly is one possible later output; an IR need not be tied to assembly generation.

C - Correct: A reusable representation of program semantics separates front-end meaning from target-specific implementation.

D - Incorrect: Control-flow visualization is one possible use and does not exhaust the purpose of IR.

Primary note proof

Study-note section COMP26

IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 11-12

4. Intermediate Code Generation
Easy to produce and easy to translate to machine code

Compiler Design module | PDF page 3 | lines 25-29

Phases of a Compiler
• Analysis
- Machine Independent/Language Dependent
• Synthesis
- Machine Dependent/Language independent

Compiler Design module | PDF page 15 | lines 13-21

Three Address Code
• Is a sequence of statements of the form
- X = Y op Z
- X,Y and Z are names, constants or compiler generated temporaries
- Op is operator (arithmetic, logical)
• Example:
- a = b + c , x = -y , if a > b goto L1
• LHS is the target
• RHS has at most two sources and one operator

Question sources

Computer Science MCQ practice collection · Question 798

Question 109 | AM0799

Which of the following is used to generate a lexical analyzer for a programming language?

A. Finite Automaton

B. Context-Free Grammar

C. Pushdown Automaton

D. Turing Machine

Hint

Follow the representation at each construction step and apply the pattern or state-set rules consistently, including empty cases.

Answer A: A finite automaton.

COMP06 describes converting regular expressions to an NFA and then a DFA. This is the automaton construction used for ordinary generated token recognizers. CFGs and pushdown machinery address nested syntax rather than the regular token-pattern mechanism; a Turing machine is not the specific scanner model in the supplied construction.

Why each choice is right or wrong

A - Correct: NFA/DFA finite automata implement recognition of regular token patterns in the stated generator pipeline.

B - Incorrect: CFGs describe nested syntactic structure rather than the regular scanner-automaton construction.

C - Incorrect: Pushdown machinery adds a stack for context-free structure and is not the ordinary token-recognizer construction.

D - Incorrect: A Turing machine is more general but is not the specific model used in this scanner-generation pipeline.

Primary note proof

Study-note section COMP06

> A common generator pipeline is regular expressions → NFA using Thompson construction → DFA using subset construction; minimization is optional. For concatenation rs, connect r's final state to s's start by ε. Alternation introduces branching ε paths; star adds a way to skip and repeat the fragment.

Study-note section COMP06

> The ε-closure of states includes those reachable by zero or more ε edges. A DFA state is a set of NFA states; its a-transition is ε-closure(move(set,a)). Start with the NFA start closure, repeatedly add unseen subsets, and mark subsets containing an NFA final state as accepting. Token identity and rule priority annotate accepting states; the empty subset acts as a dead state.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 4-9

What is Lexical Analysis
• The first phase of a compiler
• The input is a high level language program
• The output is a sequence of tokens
• Strips off blanks, tabs, newlines, and comments from the source program
• Keeps track of line numbers

Compiler Design module | PDF page 9 | lines 6-16

Grammars
• Every programming language has grammar rules
• Parsers or syntax analyzers are generated for a particular grammar
• CFG are used for syntax specification of programming languages
Context Free Grammar (CFG)
• Is denoted as G = (N, T , P, S)
• N : finite set of non-terminals
• T : finite set of terminals
• S ∈ N: The start symbol
• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Question sources

Computer Science MCQ practice collection · Question 799

Question 110 | AM0800

What is the purpose of a lexical analyzer in a compiler?

A. To check the syntax of the program

B. To generate intermediate code

C. To perform type checking

D. To recognize and tokenize the input source code

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: To recognize and tokenize the input source code

PDF p8 gives the scanner a high-level source-program input and sequence-of-tokens output.

Why each choice is right or wrong

A - Incorrect: Syntax checking belongs to parsing (p5).

B - Incorrect: Intermediate generation is separate (p6).

C - Incorrect: Type checking belongs to semantic analysis (p6).

D - Correct: Directly matches p8 input and output.

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 4-9

> What is Lexical Analysis
> • The first phase of a compiler
> • The input is a high level language program
> • The output is a sequence of tokens
> • Strips off blanks, tabs, newlines, and comments from the source program
> • Keeps track of line numbers

Question sources

Computer Science MCQ practice collection · Question 800

Question 111 | AM0801

Which listed notation directly specifies ordinary token patterns using alternation, concatenation and repetition, rather than a general production notation or a tree of one program?

A. Regular Expressions

B. Context-Free Grammars

C. Backus-Naur Form (BNF)

D. Abstract Syntax Trees (AST)

Hint

Follow the representation at each construction step and apply the pattern or state-set rules consistently, including empty cases.

Answer A: Regular Expressions

Classical regular expressions combine alternation, concatenation and repetition to specify ordinary token patterns. CFGs and BNF are production-based grammar descriptions and can also describe regular languages, so mere expressive capability would not distinguish them. An AST represents the analyzed structure of a particular program. The revised comparison asks for the direct pattern notation characterized by its operators, making regular expressions the specific answer.

Why each choice is right or wrong

A - Correct: Regular expressions directly use the stated pattern-combination operators.

B - Incorrect: A CFG is a general production grammar, not the direct expression notation described.

C - Incorrect: BNF is a notation for context-free productions.

D - Incorrect: An AST represents one analyzed program structure rather than the permitted token patterns.

Primary note proof

Study-note section COMP05

> Regular expressions use concatenation, alternation r|s, Kleene star r* for zero or more, and r+ for one or more. In a simplified alphabet, identifier = Letter(Letter|Digit)*; it permits a single letter and forbids a leading digit. A C-like ASCII rule commonly also allows underscore: [A-Za-z_][A-Za-z0-9_]*
> State the convention instead of claiming this covers every language's identifiers.

Study-note section COMP-S02

> Backus-Naur form, or BNF, is a notation for context-free grammar production rules. It describes how syntactic constructs are formed; it is different from an abstract syntax tree representing one particular program.

Study-note section COMP09

> An AST omits grammar scaffolding and punctuation; expression operands are leaves and operators internal nodes. The parser can build either representation, depending on implementation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 8 | lines 10-18

> Tokens, Patterns, and Lexemes
> • Token
> ● A string of characters which logically belong together
> ● Classes of similar lexemes
> l identifier, keywords, constants etc.
> • Pattern

- A rule which describes a token
- Lexeme
- The sequence of characters matched by a pattern to form the token

Compiler Design module | PDF page 9 | lines 6-16

Grammars
• Every programming language has grammar rules
• Parsers or syntax analyzers are generated for a particular grammar
• CFG are used for syntax specification of programming languages
Context Free Grammar (CFG)
• Is denoted as G = (N, T , P, S)
• N : finite set of non-terminals
• T : finite set of terminals
• S ∈ N: The start symbol
• P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Question sources

Computer Science MCQ practice collection · Question 801

Question 112 | AM0802

Which listed parsing approaches can handle arbitrarily nested parenthesized expressions for suitable grammars and stack-based implementations? Select all that apply.

A. LL Parsing

B. LR Parsing

C. Recursive Descent Parsing

D. Operator Precedence Parsing

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer A, B, C, D: LL Parsing; LR Parsing; Recursive Descent Parsing; Operator Precedence Parsing

Nested parentheses require remembering unfinished surrounding expressions. LL and LR parsers use a parsing stack, recursive descent uses the call stack, and suitable operator-precedence expression parsing also uses stack structure and precedence relations. Each can therefore handle the stipulated nested expressions for an appropriate grammar. The existence of nesting alone does not distinguish one of these parser families from the others.

Why each choice is right or wrong

A - Correct: LL can parse a suitable nested-expression grammar using its stack.

B - Correct: LR can recognize the nesting with bottom-up stack actions.

C - Correct: Recursive descent processes nested expressions through recursive calls.

D - Correct: A suitable operator-precedence parser supports parenthesized subexpressions.

Primary note proof

Study-note section COMP18

LL, LR and recursive-descent parsers can all handle nested parentheses for suitable grammars. An explicit or implicit stack, not the LR label alone, supports unbounded nesting.

Study-note section COMP-S01

Operator-precedence parsing is a restricted bottom-up expression-parsing method using a stack and precedence relations. With a suitable expression grammar, the stack retains unfinished surrounding expressions while nested parentheses are processed.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 18-23

Top Down Parsing
• The parse tree is created top to bottom
• Starts from the start symbol and transform it to the input
Bottom Up Parsing
• Starts with the input symbols and tries to construct the parse tree up to the start symbol.
• One way of reducing a sentence is to follow the right most derivation in reverse

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Compiler Design module | PDF page 12 | lines 17-25

Syntax Directed Definition- Example
• Production Semantic Rules
• L → E return print(E.val)
• E → E1 + T E.val = E1.val + T.val
• E → T E.val = T.val
• T → T1 * F T.val = T1.val * F.val
• T → F T.val = F.val
• T → (E) F.val = E.val
• F → digit F.val = digit.lexval

Question sources

Computer Science MCQ practice collection · Question 802

Question 113 | AM0803

Which parsing technique builds the parse tree from the bottom-up?

A. LL Parsing

B. LR Parsing

C. Recursive Descent Parsing

D. Top-Down Parsing

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: LR Parsing

LR parsing constructs the parse tree bottom-up, reducing handles that correspond to a rightmost derivation in reverse. LL methods construct from the start symbol toward the input and are top-down. Recursive descent normally implements one procedure per nonterminal, also expanding top-down. The generic top-down option states the opposite construction direction. These are implementation and derivation distinctions, not a ranking of their execution speed.

Why each choice is right or wrong

A - Incorrect: LL parsing expands a leftmost derivation from the start symbol.

B - Correct: LR parsing reduces input handles and builds bottom-up.

C - Incorrect: Recursive-descent procedures expand grammar nonterminals top-down.

D - Incorrect: Top-down construction starts from the root, the opposite direction.

Primary note proof

Study-note section COMP11

Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general

recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP15

Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

Parsing
• Process of constructing parse tree for a sentence generated by a given grammar.
• 2 types of parsers
● Top down parsing (predictive parsers)
• LL(1)
● Bottom up parsing (SR parsers)
• LR(1)

Compiler Design module | PDF page 10 | lines 18-23

Top Down Parsing
• The parse tree is created top to bottom
• Starts from the start symbol and transform it to the input
Bottom Up Parsing
• Starts with the input symbols and tries to construct the parse tree up to the start symbol.
• One way of reducing a sentence is to follow the right most derivation in reverse

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science MCQ practice collection · Question 803

Question 114 | AM0804

Which is the most specific listed name for a stack-based expression parsing method using pairwise operator-precedence relations in a suitable conflict-free operator grammar?

A. Precedence Parsing

B. Operator Precedence Parsing

C. LR Parsing

D. LL Parsing

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: Operator Precedence Parsing

Operator-precedence parsing applies precedence relations to a suitable operator grammar to choose expression parsing actions. The question specifies that concrete method rather than asking for a universal cure for shift/reduce conflicts. Precedence parsing is a broader label, while LR and LL name larger parser families without supplying these particular relations. Therefore B is the most specific match to the described technique.

Why each choice is right or wrong

A - Incorrect: Precedence parsing is broader than the specific method requested.

B - Correct: Operator-precedence parsing directly names this method.

C - Incorrect: LR alone does not specify the operator-precedence method.

D - Incorrect: LL alone does not specify the operator-precedence method.

Primary note proof

Study-note section COMP18

> An operator grammar conventionally forbids adjacent nonterminals on a RHS and usually ε productions. Operator-precedence parsing additionally requires conflict-free precedence relations.

Study-note section COMP-S01

> Operator-precedence parsing is a restricted bottom-up expression-parsing method using a stack and precedence relations. With a suitable expression grammar, the stack retains unfinished surrounding expressions while nested parentheses are processed.

Study-note section COMP11

Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science MCQ practice collection · Question 804

Question 115 | AM0805

Which listed parsing categories have a standard table-driven implementation that uses lookahead to select the next parsing action? Select all that apply. For C, mean ordinary procedure-based recursive descent with choices encoded directly in code.

A. LL Parsing

B. LR Parsing

C. Recursive Descent Parsing

D. Top-Down Parsing

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer A, B, D: LL Parsing; LR Parsing; Top-Down Parsing

A nonrecursive LL parser consults a nonterminal/lookahead table to select productions. LR parsers use ACTION and GOTO tables for shifts, reductions and state changes. Table-driven predictive parsing is also a top-down implementation, so the broader D category applies. The stipulated ordinary recursive-descent alternative encodes decisions directly in procedures instead of consulting an explicit parsing table. Thus A, B and D are selected.

Why each choice is right or wrong

A - Correct: Table-driven LL selects productions from lookahead entries.

B - Correct: LR uses ACTION/GOTO tables.

C - Incorrect: The stipulated implementation encodes decisions in procedures rather than an explicit table.

D - Correct: Nonrecursive predictive parsing is a table-driven top-down method.

Primary note proof

Study-note section COMP18

> Both LL and LR can use parsing tables.

Study-note section COMP11

> A nonrecursive predictive parser stores grammar symbols on an explicit stack and consults a table.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 11 | lines 2-17

> LL(1) Grammar
> • L - left to right
> • L - left most derivation
> • 1 - number of look ahead
> • First( ) and Follow( )
> ● the first terminal in a string and the terminal that follows a variable respectively.
> LR Parsing
> • LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
> Types of LR Parsers
> • LR (0) , SLR (1) , LALR (1) , CLR (1)
> LL LR
> Leftmost derivation Rightmost derivation in reverse
> Starts with root non-terminal on stack Ends with root non-terminal on the stack
> Builds the parse tree top-down Builds the parse tree bottom-up
> Expands the non-terminals Reduces the non-terminals

Compiler Design module | PDF page 10 | lines 11-17

Parsing
• Process of constructing parse tree for a sentence generated by a given grammar.
• 2 types of parsers
● Top down parsing (predictive parsers)
• LL(1)
● Bottom up parsing (SR parsers)
• LR(1)

Question sources

Computer Science MCQ practice collection · Question 805

Question 116 | AM0806

Which parsing technique is more suitable for handling left-recursive grammar rules?

A. LL Parsing

B. LR Parsing

C. Recursive Descent Parsing

D. Top-Down Parsing

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: LR parsing.

COMP12 states that suitable left-recursive grammars are handled by LR parsers, while ordinary unmodified recursive descent can recurse forever without consuming input. LL/top-down prediction generally needs left recursion removed first. This directly supports LR as the suitable choice, without claiming LR accepts every possible left-recursive grammar.

Why each choice is right or wrong

A - Incorrect: Ordinary LL prediction requires eliminating left recursion rather than directly exploiting it.

B - Correct: LR can directly handle suitable left-recursive grammar rules using bottom-up recognition.

C - Incorrect: Unmodified recursive descent can recurse indefinitely before consuming input on a left-recursive rule.

D - Incorrect: Ordinary top-down prediction needs recursion removal; it is not the suitable direct handling choice.

Primary note proof

Study-note section COMP12

Immediate left recursion has A→Aα. For A→Aα|β, where β does not start with A, introduce a new nonterminal A′ and use:
A→βA′
A′→αA′|ε
This preserves the generated language; tree shape changes, so semantic actions must preserve intended associativity. Indirect left recursion follows a cycle through other nonterminals and also needs removal for ordinary predictive descent.

Study-note section COMP12

Left factoring rewrites A→αβ|αγ using:
A→αA′
A′→β|γ
It postpones a choice until after the common prefix. It does not by itself remove ambiguity or guarantee LL(1). LR parsers can handle suitable left-recursive grammars; ordinary unmodified recursive descent can recurse forever before consuming input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 18-23

Top Down Parsing
• The parse tree is created top to bottom
• Starts from the start symbol and transform it to the input
Bottom Up Parsing
• Starts with the input symbols and tries to construct the parse tree up to the start symbol.
• One way of reducing a sentence is to follow the right most derivation in reverse

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science MCQ practice collection · Question 806

Question 117 | AM0807

Which of the following is not a top-down parsing algorithm?

A. Recursive Descent Parsing

B. LL Parsing

C. Predictive Parsing

D. LR Parsing

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer D: LR Parsing

Recursive descent, LL parsing and predictive parsing all expand productions from the start symbol toward the input. Predictive parsing uses lookahead to avoid backtracking, while a general recursive-descent implementation may backtrack. LR parsing instead shifts input and reduces handles to build structure bottom-up. Its construction direction therefore excludes it from the top-down family, independent of whether either implementation uses a stack.

Why each choice is right or wrong

A - Incorrect: Recursive descent expands nonterminal procedures from the start symbol.

B - Incorrect: LL constructs a leftmost derivation top-down.

C - Incorrect: Prediction chooses a top-down production using lookahead.

D - Correct: LR is bottom-up, reducing handles from the input.

Primary note proof

Study-note section COMP11

> Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP15

> Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

Parsing
• Process of constructing parse tree for a sentence generated by a given grammar.
• 2 types of parsers
● Top down parsing (predictive parsers)
• LL(1)
● Bottom up parsing (SR parsers)
• LR(1)

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science MCQ practice collection · Question 807

Question 118 | AM0808

Which listed method can retain multiple parse possibilities for a general ambiguous CFG rather than requiring an ordinary deterministic LL/LR grammar?

A. LL Parsing

B. Generalized LR parsing (GLR)

C. Recursive Descent Parsing

D. Top-Down Parsing

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: Generalized LR parsing (GLR)

A general ambiguous grammar has multiple parse trees for at least one input. Generalized LR parsing can handle those possibilities, whereas ordinary deterministic LL/LR grammars must be unambiguous. A separate disambiguation policy may still be needed to select intended meaning. Recursive descent and top-down labels alone do not provide a generalized multi-parse method, so the GLR alternative uniquely answers the question.

Why each choice is right or wrong

A - Incorrect: Ordinary deterministic LL requires an unambiguous predictive grammar.

B - Correct: GLR is the generalized method identified for ambiguous grammars.

C - Incorrect: Ordinary recursive descent alone does not specify a generalized multi-parse algorithm.

D - Incorrect: Top-down direction alone does not handle general ambiguity.

Primary note proof

Study-note section COMP18

> LL, LR and recursive-descent parsers can all handle nested parentheses for suitable grammars. An explicit or implicit stack, not the LR label alone, supports unbounded nesting. Both LL and LR can use parsing tables. Deterministic LL/LR grammars must be unambiguous; a general ambiguous grammar requires rewriting, disambiguation policy or a generalized method such as GLR.

Study-note section COMP18

> An operator grammar conventionally forbids adjacent nonterminals on a RHS and usually ε productions. Operator-precedence parsing additionally requires conflict-free precedence relations. Ordinary right-linear regular grammars also lack adjacent nonterminals, so that property alone does not uniquely identify the operator-grammar category. Regular patterns are chiefly lexical specifications; CFGs chiefly describe nested syntax.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 9 | lines 6-16

> Grammars
> • Every programming language has grammar rules
> • Parsers or syntax analyzers are generated for a particular grammar
> • CFG are used for syntax specification of programming languages
> Context Free Grammar (CFG)
> • Is denoted as G = (N, T , P, S)
> • N : finite set of non-terminals
> • T : finite set of terminals
> • S ∈ N: The start symbol
> • P : Finite set of productions, each of the form A→α, where A∈N and α ∈ (N U T)*

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science MCQ practice collection · Question 808

Question 119 | AM0809

What is the main goal of a compiler?

A. Execution of programs

B. Translation of programs

C. Optimization of programs

D. Debugging of programs

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Translation of programs

PDF p2 defines a compiler as reading a program in one language and translating it to an equivalent program in another.

Why each choice is right or wrong

A - Incorrect: Execution is not the defining translation task on p2.

B - Correct: Exactly matches the compiler definition on p2.

C - Incorrect: Optimization is an application/stage, not the defining goal (pp2,7).

D - Incorrect: Testing may use compiler technology (p2), but debugging is not its defining goal.

Primary note proof

Study-note section COMP01

> A compiler translates a source program into an equivalent target representation while preserving defined observable behavior. The target may be machine code, assembly, bytecode or another source language. A transpiler is a source-to-source compiler, not a fundamentally separate noncompiler technology. A compiler preserves language semantics, not the programmer's unstated intention.

Study-note section COMP01

> An interpreter executes a program representation; it need not work one source line at a time. Systems can compile to bytecode and then interpret or JIT-compile it. AOT compilation happens before execution; JIT compilation happens during execution and may use runtime profiles. Neither is universally faster; startup cost, optimization and workload matter.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 2 | lines 13-15

> What is Compiler
> • a program that reads a program written in one language and translates it into an equivalent program in another language.

Question sources

Computer Science MCQ practice collection · Question 809

Question 120 | AM0810

Which phase of the compiler analyzes the source code to identify the basic syntactic structures?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Syntax Analysis

PDF p5 assigns identifying expressions, statements, and declarations through grammatical grouping to syntax analysis. Here syntactic structures means those higher-level constructs.

Why each choice is right or wrong

A - Incorrect: Scanning recognizes tokens, the lexical units used by parsing (p5).

B - Correct: P5 explicitly identifies expressions, statements and declarations here.

C - Incorrect: Semantic analysis checks meaningful compatibility (p6).

D - Incorrect: Code generation translates IC to machine code (p7).

Primary note proof

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

> 2. Syntax / Hierarchical Analysis - Parsing
> • Tokens are grouped hierarchically into nested collections with collective meaning.
> • The result is generally a parse tree.
> • expressions, statements, declarations etc... are identified by using the results of lexical analysis.
> • Most syntactic errors in the source program are caught in this phase.
> • Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 5 | lines 3-11

> 1. Lexical / Linear Analysis (scanning)
> • Scans the source code as a stream of characters
> • Represent lexemes in the form of tokens as:
> <token-name, attribute-value>
> • Token
> • smallest meaningful element that a compiler understands.
> • Eg.
> • Identifiers, Keywords, Literals, Operators and Special symbols.
> • Blanks, new lines, comments will be removed from the source program.

Question sources

Computer Science MCQ practice collection · Question 810

Question 121 | AM0811

Which phase of the compiler assigns meaning to the syntactic structures and performs type checking?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: Semantic Analysis

PDF p6 explicitly checks meaningful fit and performs type checking, which identifies semantic analysis.

Why each choice is right or wrong

A - Incorrect: Lexical analysis recognizes tokens (p5).

B - Incorrect: Syntax analysis checks grammar structure (p5).

C - Correct: Matches both meaningful compatibility and type checking on p6.

D - Incorrect: Code generation converts intermediate code (p7).

Primary note proof

Study-note section COMP22

> A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

> Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

> 3. Semantic Analysis
> • Certain checks are performed to make sure that the components of the program fit

together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Computer Science MCQ practice collection · Question 811

Question 122 | AM0812

Which of the following represents the correct order of compiler phases?

A. Code Generation, Semantic Analysis, Lexical Analysis, Syntax Analysis

B. Lexical Analysis, Syntax Analysis, Semantic Analysis, Code Generation

C. Syntax Analysis, Lexical Analysis, Semantic Analysis, Code Generation

D. Semantic Analysis, Syntax Analysis, Lexical Analysis, Code Generation

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Lexical Analysis, Syntax Analysis, Semantic Analysis, Code Generation

PDF pp5-7 orders lexical, syntax, semantic, intermediate generation, optimization, and generation. B preserves the relative order of the four listed phases, though it omits two intervening stages.

Why each choice is right or wrong

A - Incorrect: Starts with generation before analysis, reversing the pp5-7 sequence.

B - Correct: Preserves the pp5-7 relative order: lexical, syntax, semantic, generation.

C - Incorrect: Syntax must consume lexical tokens (p5), so it cannot precede lexical analysis.

D - Incorrect: Semantics is phase 3 after lexical and syntax analysis (pp5-6).

Primary note proof

Study-note section COMP03

The conventional sequence is lexical analysis → syntax analysis → semantic analysis → intermediate-code generation → optimization → target-code generation. Scanning turns characters into tokens; parsing builds structure; semantics checks declarations and types; synthesis produces implementation code. A phase is a logical activity; a pass is a traversal that can combine activities.

Study-note section COMP03

The front end handles source-language rules; a middle end analyzes and transforms IR; the backend handles target details. Symbol-table management and error reporting support multiple phases rather than forming one mandatory seventh final phase. Optimization can occur at several levels, including after instruction selection. “Final phase” means code generation in the standard six-phase teaching pipeline.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 3-11

1. Lexical / Linear Analysis (scanning)
- Scans the source code as a stream of characters
- Represent lexemes in the form of tokens as:
<token-name, attribute-value>
- Token
• smallest meaningful element that a compiler understands.
- Eg.
• Identifiers, Keywords, Literals, Operators and Special symbols.
- Blanks, new lines, comments will be removed from the source program.

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 6 | lines 11-12

4. Intermediate Code Generation
Easy to produce and easy to translate to machine code

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Question sources

Computer Science MCQ practice collection · Question 812

Question 123 | AM0813

Which data structure is commonly used in symbol tables during the compilation process?

A. Stack

B. Queue

C. Hash Table

D. Linked List

Hint

Ask whether the required information is a declaration's identity and attributes, program structure or the ordered source tokens.

Answer C: A hash table.

COMP19 explicitly says a hash table offers expected constant-time insertion and lookup with suitable hashing. That makes it the conventional structure for the name-to-attribute mapping. A stack can organize nested scope tables, and lists are possible implementation choices, but the common keyed lookup structure asked for here is the hash table.

Why each choice is right or wrong

A - Incorrect: A stack manages nested scope order, while the keyed mapping within a scope commonly uses hashing.

B - Incorrect: FIFO queue order does not directly provide the keyed lookup role of a symbol table.

C - Correct: Hashing commonly implements efficient name-to-attribute insertion and lookup.

D - Incorrect: A linked list is possible but does not provide the expected constant-time keyed lookup highlighted in COMP19.

Primary note proof

Study-note section COMP19

A symbol table maps names or symbol identities to attributes: type, scope, width, storage offset, parameter signature, constant information or linkage. A hash table gives expected O(1) insertion/lookup with suitable hashing; a stack of scope tables supports lexical nesting. Search the nearest scope first, insert declarations in the current scope, and remove visibility when leaving it.

Study-note section COMP19

Scanning may intern a spelling first; semantic analysis determines the declaration and type. Implementations differ, so first interaction is conventional rather than universal. Not every punctuation token needs a symbol-table entry; literals may use a separate literal table. A hash table is an implementation, not the information's semantic role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float
id.width=4
offset = offset +id.width { offset = 6 }

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Computer Science MCQ practice collection · Question 813

Question 124 | AM0814

Which phase of the compiler performs optimization techniques to improve the efficiency of the generated code?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Optimization

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer D: Code Optimization

PDF p7 explicitly improves intermediate code by removing inefficiencies; p17 names faster execution and efficient memory as goals.

Why each choice is right or wrong

A - Incorrect: Lexical analysis tokenizes (p5).

B - Incorrect: Syntax analysis parses (p5).

C - Incorrect: Semantic analysis checks meaningful fit (p6).

D - Correct: The named improvement phase on pp7,17 is code optimization.

Primary note proof

Study-note section COMP30

> Constant folding evaluates constant expressions during compilation: 3*4→12. Constant propagation substitutes known values: x=3; y=x+2 becomes y=3+2 before folding. Copy propagation substitutes a copied name while its source remains unchanged. Dead-code elimination removes unreachable code or computations whose values are unused and whose evaluation has no necessary side effects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 2-6

> 5. Code Optimization
> Changes the IC by removing such inefficiencies
> Improve the code
> a. Improvement may be time, space, or power consumption.
> It changes the structure of programs,

Compiler Design module | PDF page 17 | lines 16-22

Chapter 8
Introduction to Code Optimization
Goals of Code Optimization
• Remove redundant code without changing the meaning of program
• Executes faster
• Efficient memory usage
• Better performance

Question sources

Computer Science MCQ practice collection · Question 814

Question 125 | AM0815

Which of the following is not a typical optimization performed by a compiler?

A. Constant Folding

B. Loop Unrolling

C. Dead Code Elimination

D. Syntax Analysis

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer D: Syntax Analysis

Constant folding evaluates known expressions, loop unrolling replicates a loop body to reduce branch overhead, and dead-code elimination removes unnecessary computation when effects permit. Each transforms an existing program while preserving its required behavior. Syntax analysis instead checks and organizes grammatical structure in the frontend. It is necessary compiler work, but it is not an optimization transformation of the kind named by the other alternatives.

Why each choice is right or wrong

A - Incorrect: Constant folding replaces constant expressions with computed values.

B - Incorrect: Unrolling replicates loop iterations, handling remainders and code-size costs.

C - Incorrect: Dead-code elimination removes unreachable or unused effect-free computations.

D - Correct: Syntax analysis recognizes grammatical structure rather than optimizing code.

Primary note proof

Study-note section COMP30

Constant folding evaluates constant expressions during compilation: 3*4→12.

Study-note section COMP30

Dead-code elimination removes unreachable code or computations whose values are unused and whose evaluation has no necessary side effects.

Study-note section COMP32

Unrolling replicates a body to reduce branch overhead and expose parallelism; handle remaining iterations and increased code size.

Study-note section COMP03

The conventional sequence is lexical analysis → syntax analysis → semantic analysis → intermediate-code generation → optimization → target-code generation. Scanning turns characters into tokens; parsing builds structure; semantics checks declarations and types; synthesis produces implementation code.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 18 | lines 2-10

Techniques
• Common sub-expression elimination
- Repeated appearance computed previously
• Strength reduction
- Replacement of expensive expressions with simple ones
• Code movement
- Moving a block of code outside a loop
• Dead code elimination
- Eliminated code statements that are either never executed or unreachable

Compiler Design module | PDF page 18 | lines 32-41

• Constant folding
- Eg.
- R2 = 3 * 2
• Constant Propagation
- Eg.
- r1 = 3
- r2 = r1 * 2
• Copy Propagation
- Eg.
- r2 = r1

Compiler Design module | PDF page 5 | lines 12-18

2. Syntax / Hierarchical Analysis - Parsing
• Tokens are grouped hierarchically into nested collections with collective meaning.
• The result is generally a parse tree.
• expressions, statements, declarations etc... are identified by using the results of lexical analysis.
• Most syntactic errors in the source program are caught in this phase.
• Syntactic rules of the source language are given via a Grammar.

Question sources

Computer Science MCQ practice collection · Question 815

Question 126 | AM0816

Which phase of the compiler generates the target code?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Code Generation

PDF p7 states that code generation converts intermediate code to machine code, the target in this question.

Why each choice is right or wrong

A - Incorrect: Lexical analysis emits tokens (p8).

B - Incorrect: Syntax analysis emits a tree (p5).

C - Incorrect: Semantic analysis checks compatibility (p6).

D - Correct: Machine-code output is the explicit code-generation role (p7).

Primary note proof

Study-note section COMP34

> Code generation lowers checked IR to a target form. Instruction selection chooses target operations; register allocation chooses where values reside; scheduling orders instructions while respecting dependencies; storage layout assigns offsets and follows ABI requirements. Register assignment names actual physical registers after allocation decisions.

Study-note section COMP34

> For t=a+b, a simple target could emit LOAD R1,a; LOAD R2,b; ADD R3,R1,R2; STORE t,R3. Other targets may combine loads with arithmetic or keep t in a register. Machine idioms, register availability and instruction order affect execution time.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Question sources

Computer Science MCQ practice collection · Question 816

Question 127 | AM0817

Which of the following is an advantage of using an intermediate representation during the compilation process?

A. Faster compilation speed

B. Improved error detection

C. Portability across different platforms

D. Reduced memory usage

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer C: Portability across platforms.

COMP26 explains that IR separates source analysis from target implementation, allowing optimizers and backends to be reused. This supports retargeting and portability across platforms. Introducing IR does not inherently promise faster compilation, better error detection or lower memory use; it can add processing stages and data structures while improving architecture.

Why each choice is right or wrong

A - Incorrect: An extra representation and processing stage do not inherently guarantee faster compilation.

B - Incorrect: Error detection depends on analyses and language rules; it is not the stated architectural advantage.

C - Correct: Separating source meaning from target lowering enables reuse across different platforms.

D - Incorrect: IR introduces storage and does not guarantee reduced compiler memory consumption.

Primary note proof

Study-note section COMP26

IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 11-12

4. Intermediate Code Generation
Easy to produce and easy to translate to machine code

Compiler Design module | PDF page 3 | lines 25-29

Phases of a Compiler
• Analysis
- Machine Independent/Language Dependent
• Synthesis
- Machine Dependent/Language independent

Compiler Design module | PDF page 7 | lines 7-11

6. Code Generation
Converts intermediate code to machine code.
Must handle all aspects of machine architecture
Storage allocation decisions are made
a. Register allocation and assignment

Question sources

Computer Science MCQ practice collection · Question 817

Question 128 | AM0818

What is the shared compiler service that supports detecting, recovering from and reporting lexical, syntax and semantic errors?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Error Handling

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer D: Error Handling

Error handling and reporting is a shared service supporting multiple compilation phases. Scanning detects malformed tokens, parsing detects invalid token arrangements, and semantic analysis detects name, scope or type violations. These are contributors rather than one phase responsible for every error category. Naming the shared service makes D the appropriate complete answer without denying the diagnostic responsibilities of the other phases.

Why each choice is right or wrong

A - Incorrect: Lexical analysis handles lexical errors but is not the complete shared service.

B - Incorrect: Syntax analysis handles parsing errors but is not the complete shared service.

C - Incorrect: Semantic analysis handles semantic errors but is not the complete shared service.

D - Correct: Error handling names the shared support service.

Primary note proof

Study-note section COMP39

> Panic-mode recovery skips input until a synchronizing token such as ; or } permits parsing to resume. Phrase-level recovery makes a local insertion/deletion. Report the location and cause while avoiding cascades of misleading diagnostics. Lexical, syntax and semantic phases each detect their own errors; error handling is a shared service.

Study-note section COMP39

> Lex/flex generates scanners from patterns and actions. Yacc means Yet Another Compiler Compiler; Yacc/Bison generates parsers from grammar rules and semantic actions. Typical specification files have definitions, %% rules, and %% user code. Classic Yacc uses LALR; Bison supports additional LR and generalized modes. Scanner generation does not replace the CFG parser.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 5 | lines 12-18

> 2. Syntax / Hierarchical Analysis - Parsing
> • Tokens are grouped hierarchically into nested collections with collective meaning.
> • The result is generally a parse tree.
> • expressions, statements, declarations etc... are identified by using the results of lexical analysis.
> • Most syntactic errors in the source program are caught in this phase.
> • Syntactic rules of the source language are given via a Grammar.

Compiler Design module | PDF page 6 | lines 2-10

> 3. Semantic Analysis
> • Certain checks are performed to make sure that the components of the program fit together meaningfully.
> • Unlike parsing, this phase checks for semantic errors in the source program (e.g. type

mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 14 | lines 16-23

Static Checking
• The compiler must check the semantic conventions of the source language
• Static Checking: ensures that certain kind of errors are detected and reported
• Example
• Type Checks: incompatible operands
• Flow Control Check
• Uniqueness Check
• Name Related Check

Question sources

Computer Science MCQ practice collection · Question 818

Question 129 | AM0819

Which of the following is true about syntax-directed translation?

A. It only involves the lexical analysis phase.

B. It involves the generation of intermediate code from the parse tree.

C. It is concerned with the optimization of the generated code.

D. It is performed during the code generation phase.

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer B: It can generate intermediate code using parse-tree structure and semantic actions.

COMP25 says syntax-directed actions may compute types, values, AST pointers or generated code. Generating IR from grammatical structure is therefore a supported use of syntax-directed translation. B describes that standard use without limiting it to lexical analysis or a single final phase. Optimization may use other analyses, and syntax-directed translation is not defined solely by optimization.

Why each choice is right or wrong

A - Incorrect: Syntax-directed translation is tied to grammatical productions and semantic actions, not solely tokenization.

B - Correct: Semantic actions associated with parsed structure can compute and emit an intermediate representation.

C - Incorrect: Optimization is not the defining task; translation actions can construct many forms of semantic output.

D - Incorrect: It is not restricted to final target-code generation; actions can run while parsing and constructing IR.

Primary note proof

Study-note section COMP25

A syntax-directed translation scheme embeds executable actions in productions; the actions run while translating a source program, not while generating the compiler. Attributes may compute types, values, AST pointers or generated code. The scheme defines an input-to-output translation, not merely an input language.

Study-note section COMP25

Create an identifier leaf with mkleaf(id,entry), a number with mkleaf(num,value), and an operator node with mknode(op,left,right). For a−4+c, construct +(−(a,4),c). Postorder traversal prints postfix: a 4 − c +. A stack evaluator pushes operands and pops right then left operands for an operator. Infix-to-postfix is useful for stack machines but not mandatory for all compilers.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 4-16

Syntax Directed Translation
• Attaching actions to the grammar rules(productions).
• Actions are executed during the compilation
- Not during the generation of the compiler
• Actions are executed according to the parsing mechanism.
Syntax Directed Definitions
• Is a generalization of a context free grammar
• Is a CFG with attributes and rules
• Attributes are associated with grammar symbols and rules with productions
• Attributes may be:
- Numbers
- Types
- Strings etc

Compiler Design module | PDF page 12 | lines 17-25

Syntax Directed Definition- Example
• Production Semantic Rules
• L → E return print(E.val)
• E → E1 + T E.val = E1.val + T.val
• E → T E.val = T.val
• T → T1 * F T.val = T1.val * F.val

• T → F T.val = F.val
• T → (E) F.val = E.val
• F → digit F.val = digit.lexval

Compiler Design module | PDF page 12 | lines 26-28

Functions for Syntax Tree Nodes
• mknode (op, left, right)
● Creates an operator node with label op &

Question sources

Computer Science MCQ practice collection · Question 819

Question 130 | AM0820

What is the general defining role of semantic rules attached to productions in a syntax-directed definition, rather than one possible specialized use?

A. To define the syntactic structure of the language.

B. To define computations of attributes such as types, values, environments and generated code.

C. To perform optimization techniques on the intermediate code.

D. To define the order of evaluation of expressions.

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer B: To define computations of attributes such as types, values, environments and generated code.

The grammar productions define syntactic structure, while semantic rules define computations of attributes associated with grammar symbols. Those attributes may represent types, values, environments or generated code. Assigning types and constraining evaluation dependencies are examples rather than exhaustive definitions. Optimization may occur in a particular translation, but is not the general defining role of all such rules.

Why each choice is right or wrong

A - Incorrect: Productions, rather than semantic rules, specify the grammatical structure.

B - Correct: Computing attributes is the general defining role.

C - Incorrect: Optimization is a possible specialized transformation, not the general definition.

D - Incorrect: Evaluation order is a specialized concern rather than the complete role.

Primary note proof

Study-note section COMP24

A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions. An annotated parse tree displays computed attribute values; it is not itself an inherited or synthesized attribute. Synthesized attributes flow from children to a parent; inherited attributes carry context from parent or siblings.

Study-note section COMP24

For E→E1+T, E.val=E1.val+T.val synthesizes an expression value. A declaration's type environment passed into its body is inherited. S-attributed definitions use only synthesized attributes and fit LR reduction-time evaluation naturally; suitable LL parsers can also compute them. L-attributed definitions restrict dependencies so a left-to-right traversal can evaluate inherited context. Attribute dependency cycles can prevent evaluation.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 12 | lines 4-16

Syntax Directed Translation
• Attaching actions to the grammar rules(productions).
• Actions are executed during the compilation
- Not during the generation of the compiler
• Actions are executed according to the parsing mechanism.
Syntax Directed Definitions
• Is a generalization of a context free grammar
• Is a CFG with attributes and rules
• Attributes are associated with grammar symbols and rules with productions
• Attributes may be:
- Numbers
- Types
- Strings etc

Compiler Design module | PDF page 12 | lines 17-25

Syntax Directed Definition- Example
• Production Semantic Rules
• L → E return print(E.val)
• E → E1 + T E.val = E1.val + T.val
• E → T E.val = T.val
• T → T1 * F T.val = T1.val * F.val
• T → F T.val = F.val
• T → (E) F.val = E.val
• F → digit F.val = digit.lexval

Compiler Design module | PDF page 14 | lines 24-30

Type Checking of Expressions
E → literal { E.type = char }
E → num { E.type = int }
E → id { E.type = lookup(id.entry) }
E→E1 mod E2 { E.type=if E1.type=int and E2.type= int

then int
else type_error }

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float
id.width=4
offset = offset +id.width { offset = 6 }

Question sources

Computer Science MCQ practice collection · Question 820

Question 131 | AM0821

Which phase of the compiler is responsible for type checking?

A. Lexical Analysis

B. Syntax Analysis

C. Semantic Analysis

D. Code Generation

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: Semantic Analysis

PDF p6 explicitly names type checking as a major task of semantic analysis.

Why each choice is right or wrong

A - Incorrect: Lexical analysis recognizes tokens (p5).

B - Incorrect: Syntax analysis checks grammatical hierarchy (p5).

C - Correct: Type checking is explicitly assigned here (p6).

D - Incorrect: Code generation converts IC to machine code (p7).

Primary note proof

Study-note section COMP22

A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Question sources

Computer Science MCQ practice collection · Question 821

Question 132 | AM0822

What is type checking in a compiler?

A. Verifying the syntax of the program.

B. Detecting lexical errors in the program.

C. Ensuring that the operations on variables are well-defined and compatible.

D. Generating the intermediate code for the program.

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: Ensuring that the operations on variables are well-defined and compatible.

PDF p14 defines type checking as enforcing type safety and lists incompatible operands; p15 makes assignment produce type_error when the identifier and expression types differ. Thus C expresses compatible operations.

Why each choice is right or wrong

A - Incorrect: Grammatical syntax is handled by parsing (p5).

B - Incorrect: Token recognition is lexical analysis (p8), not type compatibility.

C - Correct: Compatible operations and assignments match pp14-15.

D - Incorrect: Intermediate generation is distinct from semantic checking (pp6,15).

Primary note proof

Study-note section COMP22

> A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

> Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 14 | lines 2-15

> • Type Errors:
> - Improper or inconsistent operations during program execution
> • Type-safety:
> - Absence of type errors
> Type Checking
> • Semantic checks to enforce the type safety of the program
> • Semantic Checks
> - Static - done during compilation
> - Dynamic - done during run-time
> • Examples
> - Unary and binary operators
> - Number and type of arguments
> - Return statement with return type
> - Compatible assignment

Compiler Design module | PDF page 14 | lines 16-23

Static Checking
• The compiler must check the semantic conventions of the source language
• Static Checking: ensures that certain kind of errors are detected and reported
• Example
• Type Checks: incompatible operands
• Flow Control Check
• Uniqueness Check
• Name Related Check

Compiler Design module | PDF page 15 | lines 2-10

E→E1[E2] { E.type=if E2.type=int and
E1.type=array(s,t) then t else type_error }
Type Checking of Statements
S→id=E { S.type = if id.type=E.type then
void else type_error }
S→if E then S1 { S.type = if E.type=Boolean then
S1.type else type_error }
S→while E do S1 { S.type = if E.type = Boolean then
S1.type else type_error }

Question sources

Computer Science MCQ practice collection · Question 822

Question 133 | AM0823

Which of the following is an example of a static type checking technique?

A. Type inference

B. Dynamic typing

C. Run-time type checking

D. Operator overloading

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer A: Type inference.

COMP22 defines type inference as deriving types from constraints and explicitly calls it static even without explicit type annotations. Dynamic typing and runtime checks concern execution-time behavior. Operator overloading defines several meanings for an operation, but is not a replacement for the checking technique itself. Thus only type inference is the named static technique.

Why each choice is right or wrong

A - Correct: Type inference derives expression types from constraints before execution and is explicitly a static technique.

B - Incorrect: Dynamic typing does not name the requested before-execution type-inference technique.

C - Incorrect: Runtime type checking occurs during execution, so it is not static checking.

D - Incorrect: Overloading supplies operation meanings; COMP22 distinguishes it from type checking itself.

Primary note proof

Study-note section COMP22

> A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

> Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 14 | lines 2-15

> • Type Errors:
> - Improper or inconsistent operations during program execution
> • Type-safety:
> - Absence of type errors
> Type Checking
> • Semantic checks to enforce the type safety of the program
> • Semantic Checks
> - Static - done during compilation
> - Dynamic - done during run-time
> • Examples
> - Unary and binary operators
> - Number and type of arguments
> - Return statement with return type
> - Compatible assignment

Compiler Design module | PDF page 14 | lines 24-30

> Type Checking of Expressions
> E → literal { E.type = char }

E → num { E.type = int }
E → id { E.type = lookup(id.entry) }
E→E1 mod E2 { E.type=if E1.type=int and E2.type= int
then int
else type_error }

Question sources

Computer Science MCQ practice collection · Question 823

Question 134 | AM0824

Which attribute is specifically recorded for each variable in a type-oriented symbol table and distinguishes its typing information from the name that identifies the entry?

A. To store the types of variables used in the program.

B. To store the names of variables used in the program.

C. To store the intermediate code generated during parsing.

D. To store the syntax rules of the programming language.

Hint

Ask whether the required information is a declaration's identity and attributes, program structure or the ordered source tokens.

Answer A: To store the types of variables used in the program.

A symbol table associates a name or declaration identity with attributes. In a type-oriented table, the relevant attribute is the variable's type, used to check whether operations and assignments are compatible. Names identify the records but do not themselves give typing information. Intermediate instructions and grammar rules belong to other representations and are not the per-variable typing attribute requested.

Why each choice is right or wrong

A - Correct: Variable types provide the requested typing attribute.

B - Incorrect: Names identify entries but are not their type attributes.

C - Incorrect: IR instructions are not the per-variable type attribute.

D - Incorrect: Syntax rules describe the language rather than variable types.

Primary note proof

Study-note section COMP19

A symbol table maps names or symbol identities to attributes: type, scope, width, storage offset, parameter signature, constant information or linkage. A hash table gives expected O(1) insertion/lookup with suitable hashing; a stack of scope tables supports lexical nesting. Search

the nearest scope first, insert declarations in the current scope, and remove visibility when leaving it.

Study-note section COMP19

Scanning may intern a spelling first; semantic analysis determines the declaration and type. Implementations differ, so first interaction is conventional rather than universal. Not every punctuation token needs a symbol-table entry; literals may use a separate literal table. A hash table is an implementation, not the information's semantic role.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 6 | lines 2-10

3. Semantic Analysis
• Certain checks are performed to make sure that the components of the program fit together meaningfully.
• Unlike parsing, this phase checks for semantic errors in the source program (e.g. type mismatch)
- Type checking of various programming language constructs is one of the most important tasks.
• Stores type information in the symbol table or the syntax tree.
- Types of variables, function parameters, array dimensions, etc.

Compiler Design module | PDF page 17 | lines 2-15

Declarations
• Involves allocation of space in memory &
• Entry of type and name in symbol table
• Off set variable (Offset=0) is used to denote the base address
int a; float b;
Allocation process: { offset = 0 }
int a;
id.type = int
id.width = 2
offset = offset + id.width { offset = 2 }
float b;
id.type=float
id.width=4
offset = offset +id.width { offset = 6 }

Compiler Design module | PDF page 14 | lines 24-30

Type Checking of Expressions
E → literal { E.type = char }
E → num { E.type = int }
E → id { E.type = lookup(id.entry) }
E→E1 mod E2 { E.type=if E1.type=int and E2.type= int
then int
else type_error }

Question sources

Computer Science MCQ practice collection · Question 824

Question 135 | AM0825

In ordinary C++ with no custom conversions, which offered example is specifically a type mismatch rather than a syntax, value-domain or layout issue?

A. Syntax error in an if-else statement.

B. Division of a number by zero.

C. int x="hello";

D. Incorrect indentation in the program.

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: int x="hello";

An ordinary C++ declaration int x="hello" attempts to initialize an integer from a string literal and is a type mismatch under the stated rules. Division by zero can involve correctly typed operands and is instead a value-domain fault. Incorrect indentation is layout, while malformed if-else syntax concerns grammatical arrangement. The option illustrates an incompatible-type example.

Why each choice is right or wrong

A - Incorrect: This explicitly describes a syntax error rather than a type mismatch.

B - Incorrect: A zero divisor is a value issue even with compatible numeric types.

C - Correct: A string literal is not a compatible ordinary integer initializer.

D - Incorrect: Indentation does not establish an incompatible operand type.

Primary note proof

Study-note section COMP22

> A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

> Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 14 | lines 2-15

• Type Errors:
- Improper or inconsistent operations during program execution
• Type-safety:
- Absence of type errors

Type Checking
• Semantic checks to enforce the type safety of the program
• Semantic Checks
- Static - done during compilation
- Dynamic - done during run-time
• Examples
- Unary and binary operators
- Number and type of arguments
- Return statement with return type
- Compatible assignment

Compiler Design module | PDF page 14 | lines 24-30

Type Checking of Expressions
E → literal { E.type = char }
E → num { E.type = int }
E → id { E.type = lookup(id.entry) }
E→E1 mod E2 { E.type=if E1.type=int and E2.type= int
then int
else type_error }

Compiler Design module | PDF page 15 | lines 2-10

E→E1[E2] { E.type=if E2.type=int and
E1.type=array(s,t) then t else type_error }
Type Checking of Statements
S→id=E { S.type = if id.type=E.type then
void else type_error }
S→if E then S1 { S.type = if E.type=Boolean then
S1.type else type_error }
S→while E do S1 { S.type = if E.type = Boolean then
S1.type else type_error }

Question sources

Computer Science MCQ practice collection · Question 825

Question 136 | AM0826

Which advantage of static type checking specifically concerns detecting incompatible operations before the program executes?

A. Improved program performance.

B. Enhanced code optimization.

C. Early detection of potential runtime errors.

D. Simplified debugging process.

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: Early detection of potential runtime errors.

Static type checking can reject an incompatible operation during compilation, before that operation causes a run-time type error. This is the direct timing benefit asked for. Type information may also aid optimization, and diagnostics may make debugging easier, but those are additional consequences rather than the specific before-execution detection property. Type checking does not by itself guarantee faster code or eliminate all possible run-time errors.

Why each choice is right or wrong

A - Incorrect: Faster execution is not guaranteed and does not state detection before execution.

B - Incorrect: Optimization may use types but is a different benefit from the requested early rejection.

C - Correct: Compile-time rejection detects incompatible operations before execution.

D - Incorrect: Debugging may be easier, but that broad consequence does not specify the requested detection timing.

Primary note proof

Study-note section COMP22

> A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

> Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 14 | lines 2-15

• Type Errors:
- Improper or inconsistent operations during program execution
• Type-safety:
- Absence of type errors

Type Checking
• Semantic checks to enforce the type safety of the program
• Semantic Checks
- Static - done during compilation
- Dynamic - done during run-time
• Examples
- Unary and binary operators
- Number and type of arguments
- Return statement with return type
- Compatible assignment

Compiler Design module | PDF page 14 | lines 16-23

Static Checking
• The compiler must check the semantic conventions of the source language
• Static Checking: ensures that certain kind of errors are detected and reported
• Example
• Type Checks: incompatible operands
• Flow Control Check
• Uniqueness Check
• Name Related Check

Question sources

Computer Science MCQ practice collection · Question 826

Question 137 | AM0827

What is the primary goal of type checking in a compiler?

A. To ensure syntactic correctness of the program.

B. To generate efficient machine code.

C. To enforce programming language rules and constraints.

D. To provide support for code reuse through libraries.

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: To enforce programming language rules and constraints.

PDF p14 defines type checking as semantic checks enforcing type safety and says the compiler checks source-language semantic conventions. C is the offered broad rule-enforcement description, limited here to type constraints.

Why each choice is right or wrong

A - Incorrect: Syntax correctness is the parsing task (p5).

B - Incorrect: Machine-code efficiency is an optimization goal (pp7,17).

C - Correct: Type-related semantic conventions and constraints match p14.

D - Incorrect: Library reuse is not a type-checking goal stated in the module.

Primary note proof

Study-note section COMP22

> A type checker verifies that operations are defined for their operands. In a simple strict language, integer mod requires two integers; array indexing requires an array and integer index; an if condition requires Boolean; assignment requires equal or convertible types. Function calls check parameter count and compatibility, and returns match result types.

Study-note section COMP22

> Type inference derives types from constraints: if x is integer and + expects numeric operands, infer the expression type from numeric promotion rules. It is a static technique even without explicit annotations. Operator overloading supplies multiple operation meanings, not a substitute for type checking. int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 14 | lines 2-15

> • Type Errors:
> - Improper or inconsistent operations during program execution
> • Type-safety:
> - Absence of type errors
> Type Checking
> • Semantic checks to enforce the type safety of the program
> • Semantic Checks
> - Static - done during compilation
> - Dynamic - done during run-time
> • Examples
> - Unary and binary operators
> - Number and type of arguments
> - Return statement with return type
> - Compatible assignment

Compiler Design module | PDF page 14 | lines 16-23

Static Checking
• The compiler must check the semantic conventions of the source language
• Static Checking: ensures that certain kind of errors are detected and reported
• Example
• Type Checks: incompatible operands
• Flow Control Check
• Uniqueness Check
• Name Related Check

Question sources

Computer Science MCQ practice collection · Question 827

Question 138 | AM0828

Which of the following is an example of a type coercion or type conversion?

A. Assigning a floating-point value to an integer variable.

B. Assigning a value to an uninitialized variable.

C. Declaring a variable without specifying its type.

D. Using an undeclared variable in the program.

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer A: Converting a floating-point value when assigning to an integer variable, where the language permits it.

COMP23 gives C++ int x=3.14 as an ordinary conversion yielding 3 when representable. That is the listed example of conversion, with the language caveat that Java needs a permitted cast and C++ brace initialization rejects narrowing. The remaining choices concern initialization, inference or undeclared names rather than converting one value type to another.

Why each choice is right or wrong

A - Correct: Floating-to-integer assignment performs conversion where permitted; exact legality and precision loss depend on the language.

B - Incorrect: First assigning to a variable is initialization or assignment, not inherently a change of value type.

C - Incorrect: Omitting an explicit declaration type may invoke inference; it does not itself convert a value.

D - Incorrect: An undeclared name is a name-resolution problem rather than an example of value conversion.

Primary note proof

Study-note section COMP23

Coercion is an implicit conversion permitted by the language; a cast requests an explicit conversion. Widening often increases representable range, but can still lose precision: Java int→float is widening although large integers round. Java int→long is exact. Narrowing can lose range or fractional information.

Study-note section COMP23

In C++ ordinary int x=3.14 converts to 3 when representable; it is not inherently a type error. int x{3.14} is rejected as narrowing. C++ floating-to-integer conversion discards the fraction and has undefined behavior if the truncated value is unrepresentable. Java ordinary double→int assignment needs a permitted explicit cast. Mixing char and int is ordinarily valid after promotion in C++/Java; do not import a fictitious universal ban.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 14 | lines 2-15

• Type Errors:
- Improper or inconsistent operations during program execution
• Type-safety:
- Absence of type errors

Type Checking
• Semantic checks to enforce the type safety of the program
• Semantic Checks
- Static - done during compilation
- Dynamic - done during run-time
• Examples
- Unary and binary operators
- Number and type of arguments
- Return statement with return type
- Compatible assignment

Compiler Design module | PDF page 15 | lines 2-10

E→E1[E2] { E.type=if E2.type=int and
E1.type=array(s,t) then t else type_error }
Type Checking of Statements
S→id=E { S.type = if id.type=E.type then
void else type_error }
S→if E then S1 { S.type = if E.type=Boolean then
S1.type else type_error }
S→while E do S1 { S.type = if E.type = Boolean then
S1.type else type_error }

Question sources

Computer Science MCQ practice collection · Question 828

Question 139 | CS0121

Which listed approaches are bottom-up and therefore are not top-down parsers? Select all that apply. Assume predictive recursive descent for A.

A. Recursive-descent parser

B. Shift-reduce parser

C. Non-recursive predictive parser

D. LR parser

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B, D: Shift-reduce parser; LR parser

Shift-reduce parsing starts from the input and reduces recognized handles toward the start symbol. LR is a bottom-up family using that strategy. Recursive descent and nonrecursive prediction instead expand productions top-down for suitable grammars. Therefore both B and D fail the top-down description. This directional distinction avoids an unsupported efficiency ranking between recursive and table-driven predictive implementations.

Why each choice is right or wrong

A - Incorrect: Predictive recursive descent is top-down.

B - Correct: Shift-reduce is bottom-up.

C - Incorrect: Nonrecursive prediction is top-down.

D - Correct: LR is bottom-up.

Primary note proof

Study-note section COMP11

> Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP15

> Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 10 | lines 11-17

Parsing
• Process of constructing parse tree for a sentence generated by a given grammar.
• 2 types of parsers
● Top down parsing (predictive parsers)
• LL(1)
● Bottom up parsing (SR parsers)
• LR(1)

Compiler Design module | PDF page 11 | lines 2-17

LL(1) Grammar
• L - left to right
• L - left most derivation
• 1 - number of look ahead
• First( ) and Follow( )
● the first terminal in a string and the terminal that follows a variable respectively.
LR Parsing
• LR(k) - Left to right scanning with Rightmost derivation in reverse, k being the number of lookahead tokens.
Types of LR Parsers
• LR (0) , SLR (1) , LALR (1) , CLR (1)
LL LR
Leftmost derivation Rightmost derivation in reverse
Starts with root non-terminal on stack Ends with root non-terminal on the stack
Builds the parse tree top-down Builds the parse tree bottom-up
Expands the non-terminals Reduces the non-terminals

Question sources

Computer Science Exit Exam practice collection · Question 121

Question 140 | CS0267

Which one of the following is a type of compiler that compiles only the changed lines from source code and updates the object code accordingly?

A. Retargetable compiler

B. One pass compiler

C. Incremental compiler

D. Parallelizing compiler

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer C: An incremental compiler.

COMP02 defines incremental compilation as recompiling affected fragments or dependent units after edits. This is the compiler category intended by updating code after source changes. The question's only changed lines wording is simplified: real dependency tracking can require recompiling more than literal edited lines. Retargeting and one-pass organization describe different properties.

Why each choice is right or wrong

A - Incorrect: Retargetable compilers support changing target descriptions rather than selecting edited source fragments.

B - Incorrect: One pass counts traversals of the program, not whether edits trigger selective recompilation.

C - Correct: Incremental compilation updates affected fragments after edits, accounting for dependencies beyond literal changed lines.

D - Incorrect: Parallelizing compilation is not the category defined by selective recompilation after source changes.

Primary note proof

Study-note section COMP02

> A preprocessor handles directives such as file inclusion and macros. An assembler translates assembly to object code. A linker combines object files and libraries, resolves external symbols and performs relocation. A loader maps an executable into memory and starts it. Compilation alone does not necessarily produce a ready executable.

Study-note section COMP02

> A cross-compiler runs on a host platform and generates code for a different target. Multiple targets require appropriate backends, not a special definition of cross-compilation. A retargetable compiler supports changing target descriptions. Incremental compilation recompiles affected fragments or dependent units after edits; it is more than literally replacing changed lines. One-pass versus multipass counts traversals, not supported targets.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 2 | lines 13-15

> What is Compiler
> • a program that reads a program written in one language and translates it into an equivalent program in another language.

Compiler Design module | PDF page 3 | lines 20-24

> • Cross-Compiler
> • compiler that runs on platform (A) and generates executable code for another platform (B).

- Source-to-source Compiler
- compiler that translates source code of one programming language to another

Question sources

Computer Science Exit Exam practice collection · Question 267

Question 141 | CS0268

Which technique of code optimization shifts computations from run time to compile time by evaluating constant expressions at compile time and replacing them with their values?

A. Copy propagation

B. Compile time evaluation

C. Code motion

D. Dead code elimination

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer B: Compile time evaluation

PDF p18 lists constant folding and illustrates R2 = 3 * 2. Evaluating this constant expression early gives 6, eliminating that runtime computation. B describes this operation as compile time evaluation, although constant folding would be the module's exact name.

Why each choice is right or wrong

A - Incorrect: P18 copy propagation example copies r1 into r2, not evaluates a constant arithmetic expression.

B - Correct: Evaluating 3 * 2 to 6 is the constant-folding example on p18, matching the stem.

C - Incorrect: P18 code movement relocates a block outside a loop, not necessarily evaluating it at compilation.

D - Incorrect: P18 dead-code elimination removes unreachable statements, rather than computing constant values.

Primary note proof

Study-note section COMP30

Constant folding evaluates constant expressions during compilation: 3*4→12. Constant propagation substitutes known values: x=3; y=x+2 becomes y=3+2 before folding. Copy propagation substitutes a copied name while its source remains unchanged. Dead-code elimination removes unreachable code or computations whose values are unused and whose evaluation has no necessary side effects.

Module passages - related reading

These source passages provide context; the primary note proof is identified separately above.

Compiler Design module | PDF page 18 | lines 32-41

- Constant folding
  - Eg.
  - R2 = 3 * 2
- Constant Propagation
  - Eg.
  - r1 = 3
  - r2 = r1 * 2
- Copy Propagation
  - Eg.
  - r2 = r1

Compiler Design module | PDF page 18 | lines 2-10

Techniques
- Common sub-expression elimination
  - Repeated appearance computed previously
- Strength reduction
  - Replacement of expensive expressions with simple ones
- Code movement
  - Moving a block of code outside a loop
- Dead code elimination
  - Eliminated code statements that are either never executed or unreachable

Question sources

Computer Science Exit Exam practice collection · Question 268

Question 142 | NS0222

For const int x=8; const int y=16; if(x>y){int z=x+y;}, which techniques directly evaluate the known condition and remove its unreachable body? Select all that apply. Arithmetic simplification means algebraic identity rewriting here.

A. Code motion

B. Dead code elimination

C. Compile time evaluation

D. Arithmetic simplification

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer B, C: Dead code elimination; Compile time evaluation

Both x and y are known constants, so evaluating 8>16 at compile time yields false. The body can never execute and dead-code elimination can remove it. These are distinct but compatible transformations, corresponding to B and C in this option order. Code motion relocates computations, while algebraic identity simplification is not the decisive operation on this constant-false condition.

Why each choice is right or wrong

A - Incorrect: No relocation is needed to evaluate or delete the branch.

B - Correct: The unreachable branch body can be removed.

C - Correct: The condition is evaluated from known constants at compile time.

D - Incorrect: No algebraic identity rewrite is the decisive step.

Primary note proof

Study-note section COMP30

> For const int x=8,y=16; if(x>y){int z=x+y;}, propagate the constants, fold 8>16 to false, then remove the unreachable branch. Both compile-time evaluation and dead-code elimination apply.

Study-note section COMP31

> Algebraic simplification removes identities when valid: integer x+0→x.

Question sources

Computer Science Exit Exam practice collection · Question 263

Question 143 | NS0223

What is the difference between a compiler and an interpreter?

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer: A compiler translates; an interpreter executes a representation.

COMP01 distinguishes translating source into an equivalent target from executing a program representation. Bytecode systems may combine compilation, interpretation and JIT, so the distinction is not simply all-at-once versus line-by-line execution.

Primary note proof

Study-note section COMP01

> A compiler translates a source program into an equivalent target representation while preserving defined observable behavior. The target may be machine code, assembly, bytecode or another source language. A transpiler is a source-to-source compiler, not a fundamentally separate noncompiler technology. A compiler preserves language semantics, not the programmer's unstated intention.

Study-note section COMP01

> An interpreter executes a program representation; it need not work one source line at a time. Systems can compile to bytecode and then interpret or JIT-compile it. AOT compilation happens before execution; JIT compilation happens during execution and may use runtime profiles. Neither is universally faster; startup cost, optimization and workload matter.

Question sources

Compiler Design review questions · Question 1

Question 144 | NS0224

What is the difference between lexical analysis and syntax analysis?

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer: Lexical analysis groups characters into tokens; syntax analysis groups tokens into structures.

COMP41 separates character-to-token classification from token-to-expression or statement structure. The scanner recognizes lexemes such as identifiers and operators and emits an ordered token stream. The parser checks how those tokens form grammatical constructs and can build a parse tree or AST; legal tokens can still appear in an illegal syntactic arrangement.

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input. An AST omits grammar scaffolding and punctuation; expression operands are leaves and operators internal nodes. The parser can build either representation, depending on implementation.

Question sources

Compiler Design review questions · Question 2

Question 145 | NS0225

What is the purpose of a symbol table?

Hint

Ask whether the required information is a declaration’s identity and attributes, program structure or the ordered source tokens.

Answer: A symbol table associates names with declaration attributes.

COMP19 maps names or identities to type, scope, width, offsets, signatures and linkage information. These records support declaration lookup and semantic checks; a hash table is one implementation rather than the purpose itself.

Primary note proof

Study-note section COMP19

> A symbol table maps names or symbol identities to attributes: type, scope, width, storage offset, parameter signature, constant information or linkage. A hash table gives expected O(1) insertion/lookup with suitable hashing; a stack of scope tables supports lexical nesting. Search the nearest scope first, insert declarations in the current scope, and remove visibility when leaving it.

Study-note section COMP19

> Scanning may intern a spelling first; semantic analysis determines the declaration and type. Implementations differ, so first interaction is conventional rather than universal. Not every punctuation token needs a symbol-table entry; literals may use a separate literal table. A hash table is an implementation, not the information’s semantic role.

Question sources

Compiler Design review questions · Question 3

Question 146 | NS0226

What is the difference between top-down and bottom-up parsing?

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer: Top-down expands from the start symbol; bottom-up reduces input toward it.

COMP15 describes bottom-up recognition as input reductions toward S, reversing a rightmost derivation. COMP11 describes top-down expansion from S toward the input, normally a leftmost derivation.

Primary note proof

Study-note section COMP11

> Top-down parsing starts with S and expands productions toward the input, usually constructing a leftmost derivation. Recursive descent typically has one procedure per nonterminal. A general recursive-descent implementation can backtrack; a predictive parser chooses a production using lookahead without backtracking. LL(1) scans Left to right, produces a Leftmost derivation and uses one lookahead token.

Study-note section COMP15

> Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse. A handle is a production RHS occurrence whose reduction is the next step of that reverse derivation. Not every substring matching a RHS is a handle in the current context.

Question sources

Compiler Design review questions · Question 4

Question 147 | NS0227

What is intermediate code and why is it used?

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer: IR separates source analysis from target lowering and enables reuse.

COMP26 defines IR's role in separating source meaning from target implementation so analysis, optimization and backends can be reused. ASTs, graphs, TAC and bytecode are possible forms. A useful IR is easy to generate, analyze, transform and lower; portability does not require every IR to exclude all target-specific details.

Primary note proof

Study-note section COMP26

> IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

> Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Question sources

Compiler Design review questions · Question 5

Question 148 | NS0228

Which of the following is NOT a valid token type produced during lexical analysis?

A. Keyword |

B. Parse tree node

C. Identifier |

D. Numeric literal

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Parse-tree node is not a lexical token type.

COMP04 lists keywords, identifiers and literals among token classes. A parse-tree node belongs to the grammatical structure built after scanning, not the lexical class of a source lexeme. Consequently only parse-tree node is the invalid token-type choice, while the other three are ordinary scanner output categories.

Why each choice is right or wrong

A - Incorrect: Keywords are explicitly listed as a valid lexical token class.

B - Correct: Parse-tree nodes represent grammatical structure, not a lexical token category emitted by a scanner.

C - Incorrect: Identifiers are an ordinary lexical token class.

D - Incorrect: Numeric literals are literals and therefore valid token categories.

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Question sources

Compiler Design review questions · Question 1

Question 149 | NS0229

What algorithm is commonly used to convert a Non-deterministic Finite Automaton (NFA) to a Deterministic Finite Automaton (DFA) in lexer construction?

A. Thompson's construction |

B. Subset construction (Powerset construction)

C. Cocke-Younger-Kasami (CYK) |

D. Earley's algorithm

Hint

Follow the representation at each construction step and apply the pattern or state-set rules consistently, including empty cases.

Answer B: Subset (powerset) construction.

COMP06 distinguishes Thompson construction, which builds an NFA from a regular expression, from subset construction, which determinizes the NFA. Each DFA state represents an NFA-state subset, with epsilon closure when needed. CYK and Earley are grammar-parsing algorithms rather than this automaton determinization, so the powerset option is the relevant stage.

Why each choice is right or wrong

A - Incorrect: Thompson construction goes from regular expression to NFA rather than NFA to DFA.

B - Correct: Subset construction creates DFA states from sets of NFA states.

C - Incorrect: CYK performs CFG membership parsing and does not determinize a lexical NFA.

D - Incorrect: Earley is a grammar-parsing method rather than the stated NFA-to-DFA construction.

Primary note proof

Study-note section COMP06

> A common generator pipeline is regular expressions → NFA using Thompson construction → DFA using subset construction; minimization is optional.

Study-note section COMP06

> The ε-closure of states includes those reachable by zero or more ε edges. A DFA state is a set of NFA states; its a-transition is ε-closure(move(set,a)). Start with the NFA start closure, repeatedly add unseen subsets, and mark subsets containing an NFA final state as accepting. Token identity and rule priority annotate accepting states; the empty subset acts as a dead state.

Study-note section COMP-S02

> CYK and Earley are context-free grammar parsing algorithms. They determine grammatical structure or membership of an input sequence. They are different tasks from Thompson construction, which builds a token-recognizing NFA from a regular expression, and subset construction, which turns an NFA into a DFA.

Question sources

Compiler Design review questions · Question 2

Question 150 | NS0230

Given the grammar: E → E + T | T, T → T * F | F, F → (E) | id - this grammar enforces:

A. Right-associativity for both + and * |

B. Left-associativity for both + and *, with * having higher precedence than +

C. Right-associativity for + and left-associativity for * |

D. No precedence or associativity rules

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Both operators associate left; multiplication has higher precedence.

COMP10 states exactly these productions and their effect. Multiplication occurs within T below the addition level E, so it binds more tightly. The recursive operand on the left of E→E+T and T→T*F enforces left association for each operator. Parentheses in F can override the default grouping.

Why each choice is right or wrong

A - Incorrect: The recursive nonterminal is on the left, producing left rather than right association.

B - Correct: Separate T multiplication and E addition levels give higher multiplication precedence and left association.

C - Incorrect: Addition is left-recursive too, so it does not have the claimed right association.

D - Incorrect: The grammar explicitly encodes both precedence levels and associativity.

Primary note proof

Study-note section COMP10

> Use E→E+T|T, T→T*F|F, F→(E)|id to give * higher precedence and both operators left associativity. For subtraction, a−b−c then groups (a−b)−c.

Question sources

Compiler Design review questions · Question 3

Question 151 | NS0231

For a reduced CFG (all symbols reachable and productive), which conditions prevent it from being LL(1)? Select all that apply. In A, overlap means a shared terminal in competing RHS FIRST sets.

A. Two productions for the same non-terminal have overlapping FIRST sets |

B. The grammar contains a left-recursive rule

C. The grammar is ambiguous |

D. A and B only

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer A, B, C: Two productions for the same non-terminal have overlapping FIRST sets |; The grammar contains a left-recursive rule; The grammar is ambiguous |

A shared FIRST terminal for alternatives of one nonterminal creates a predictive-table collision. Reachable left recursion prevents ordinary LL(1) prediction, and an ambiguous useful grammar cannot have a unique LL(1) parse. Thus A, B and C are all obstructions. The combined D alternative wrongly says only A and B, excluding ambiguity, so it is not a complete correct selection.

Why each choice is right or wrong

A - Correct: Shared FIRST terminals create competing table entries.

B - Correct: Reachable left recursion rules out ordinary LL(1) parsing.

C - Correct: A useful ambiguous grammar is not LL(1).

D - Incorrect: The word only wrongly excludes the valid ambiguity obstruction.

Primary note proof

Study-note section COMP14

> For every A→α, put it in M[A,a] for each terminal a in FIRST(α). If α is nullable, also enter it under every symbol in FOLLOW(A). More than one production in a cell is a conflict. Pairwise SELECT sets must be disjoint; SELECT(α)=FIRST(α)−{ε}, plus FOLLOW(A) when α is nullable. This also detects two nullable alternatives.

Study-note section COMP12

> Indirect left recursion follows a cycle through other nonterminals and also needs removal for ordinary predictive descent.

Study-note section COMP12

> LR parsers can handle suitable left-recursive grammars; ordinary unmodified recursive descent can recurse forever before consuming input.

Study-note section COMP18

> Deterministic LL/LR grammars must be unambiguous; a general ambiguous grammar requires rewriting, disambiguation policy or a generalized method such as GLR.

Question sources

Compiler Design review questions · Question 4

Question 152 | NS0232

Which of the following is an example of a synthesized attribute in attribute grammars?

A. The type of an expression computed from its sub-expressions |

B. The scope environment passed down to a block

C. The inherited type context for a function body |

D. The enclosing loop level passed to a break statement

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer A: Expression type computed from subexpressions is synthesized.

COMP24 defines synthesized attributes by flow from children to a parent. Computing an expression's type from its subexpressions fits this direction. Scope, type context and enclosing-loop information passed down from surrounding constructs instead supply inherited context. The direction of dependency, rather than the fact that types are involved, distinguishes these choices.

Why each choice is right or wrong

A - Correct: The parent expression derives its type from child subexpressions, which is synthesized upward flow.

B - Incorrect: A scope environment passed down to a block is inherited contextual information.

C - Incorrect: A type context supplied to a function body flows from its surroundings, so it is inherited.

D - Incorrect: Enclosing-loop context is passed inward to a break statement rather than synthesized from its children.

Primary note proof

Study-note section COMP24

> A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions. An annotated parse tree displays computed attribute values; it is not itself an inherited or synthesized attribute. Synthesized attributes flow from children to a parent; inherited attributes carry context from parent or siblings.

Study-note section COMP24

For E→E1+T, E.val=E1.val+T.val synthesizes an expression value. A declaration's type environment passed into its body is inherited. S-attributed definitions use only synthesized attributes and fit LR reduction-time evaluation naturally; suitable LL parsers can also compute them. L-attributed definitions restrict dependencies so a left-to-right traversal can evaluate inherited context. Attribute dependency cycles can prevent evaluation.

Question sources

Compiler Design review questions · Question 5

Question 153 | NS0233

What is the primary purpose of Three-Address Code (TAC) as an intermediate representation?

A. To directly execute on the target CPU |

B. To provide a machine-independent representation suitable for optimization and code generation

C. To replace the symbol table |

D. To encode only arithmetic expressions

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer B: An optimization-friendly intermediate representation for later target generation.

COMP26 gives reuse and separation as the purpose of IR and describes TAC as simple instruction records. TAC is normally not directly executed by a native CPU, does not replace declaration metadata and is not limited to arithmetic: branches and assignments are also representable. B states its architectural optimization/code-generation role most directly.

Why each choice is right or wrong

A - Incorrect: TAC is normally intermediate and requires lowering rather than direct execution on the target CPU.

B - Correct: TAC provides analyzable intermediate operations supporting reuse, optimization and later target-code generation.

C - Incorrect: A symbol table retains declaration metadata and is not replaced by the instruction representation.

D - Incorrect: TAC can encode control transfers and other operations, not only arithmetic expressions.

Primary note proof

Study-note section COMP26

IR separates source analysis from target implementation, allowing optimizers and backends to be reused. It should be easy to generate, analyze, transform and lower.

Study-note section COMP26

Ordinary TAC arithmetic has at most one operation, two sources and one destination: t1=c*d; t2=b+t1; a=t2. Quadruples store (op,arg1,arg2,result); triples omit an explicit result field and refer to instruction positions. Indirect triples reference a separate ordering list, easing rearrangement. Trees describe expressions; DAGs share common subexpressions, while quadruples/triples are instruction-record structures.

Question sources

Compiler Design review questions · Question 6

Question 154 | NS0234

The 'maximal munch' rule in lexical analysis means:

A. The lexer always produces the shortest possible token |

B. The lexer always produces the longest possible token matching the current position

C. Keyword tokens are prioritized over identifiers |

D. Comments are included in the token stream

Hint

Follow the representation at each construction step and apply the pattern or state-set rules consistently, including empty cases.

Answer B: Choose the longest matching lexeme at the current position.

COMP07 defines longest matching lexeme selection at each input position. Equal-length matches use rule order, which is a separate tie-breaking rule; keywords do not override a longer identifier such as ifx. Comments are normally skipped rather than retained merely because maximal munch is used. This establishes B as the definition.

Why each choice is right or wrong

A - Incorrect: Shortest-match selection is the opposite of maximal munch.

B - Correct: The lexer consumes the longest available match beginning at the current input position.

C - Incorrect: Keyword priority can break equal-length rule ties but does not define longest match.

D - Incorrect: Comment retention is independent of longest match; ordinary comments are skipped.

Primary note proof

Study-note section COMP07

> At each input position, choose the longest matching lexeme; if equal lengths match multiple flex rules, choose the earliest rule. Thus ifx is an identifier rather than if followed by x; <= and ++ are each one operator token. An implementation may recognize an identifier first and then consult a keyword table.

Question sources

Compiler Design review questions · Question 7

Question 155 | NS0235

In bottom-up parsing, a 'shift-reduce conflict' occurs when:

A. There are two possible reductions for the same stack top |

B. The parser cannot decide whether to shift the next input token or reduce the current stack contents

C. The grammar is left-recursive |

D. The FOLLOW set is empty

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: A competing shift and reduction on the same situation.

COMP17 defines a shift/reduce conflict as a choice between consuming a token and reducing. Two competing reductions describe reduce/reduce instead. Left recursion may be entirely acceptable to LR, and an empty FOLLOW set is not the definition of this table conflict. The conflict concerns competing actions for the same state and lookahead.

Why each choice is right or wrong

A - Incorrect: Two possible reductions define a reduce/reduce conflict, a different conflict type.

B - Correct: The same parser situation permits both shifting input and reducing the recognized right-hand side.

C - Incorrect: LR can handle suitable left recursion; that property does not itself define a shift/reduce conflict.

D - Incorrect: FOLLOW-set emptiness does not express competing shift and reduction actions.

Primary note proof

Study-note section COMP17

> SLR(1) uses LR(0) states and FOLLOW(A) for reduction by a completed A production. Canonical LR(1) attaches context-specific lookahead to items. LALR(1) merges canonical states with the same LR(0) core, combining lookaheads; merging can introduce reduce/reduce conflicts. The standard grammar-class hierarchy is LR(0)⊆SLR(1)⊆LALR(1)⊆LR(1), with increasing capability and often table size.

Study-note section COMP17

> A shift/reduce conflict offers both consuming a token and reducing; reduce/reduce offers two reductions. An SLR or LALR conflict does not by itself prove the grammar ambiguous. Canonical LR(1) handles LR(1) grammars, not every unambiguous CFG. Precedence and associativity can resolve suitable expression conflicts; changing parser names alone does not repair all ambiguity.

Question sources

Compiler Design review questions · Question 8

Question 156 | NS0236

Which optimization technique moves loop-invariant computations outside the loop body?

A. Strength reduction |

B. Loop unrolling

C. Loop-invariant code motion (LICM) |

D. Dead code elimination

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer C: Loop-invariant code motion.

COMP32 explains that an expression whose inputs remain unchanged can be moved to a preheader when safe on newly executed paths. That is loop-invariant code motion. Strength reduction changes operation cost, unrolling replicates the body, and dead-code elimination removes unused or unreachable work. Hoisting also requires safety, not merely unchanged operands.

Why each choice is right or wrong

A - Incorrect: Strength reduction substitutes a cheaper equivalent operation rather than necessarily moving it outside a loop.

B - Incorrect: Unrolling replicates loop bodies instead of hoisting invariant computations.

C - Correct: LICM moves invariant computations to an appropriate outside location when executing them there is safe.

D - Incorrect: Dead-code elimination removes unnecessary computations rather than relocating a needed invariant value.

Primary note proof

Study-note section COMP32

> A loop-invariant expression has unchanged inputs during a loop. Hoist it to a preheader only when evaluation is safe on the paths where it will newly run. Induction variables change by a loop-invariant increment, such as i=i+1. For repeated t=4*i, maintain a running t increased by 4 when equivalent, reducing repeated multiplication.

Study-note section COMP32

> Loop fusion or jamming combines loops with compatible bounds and safe dependence order. Unrolling replicates a body to reduce branch overhead and expose parallelism; handle remaining iterations and increased code size. Loop-invariant code motion, fusion, unrolling and strength reduction are different transformations. A finite taxonomy needs a criterion: machine dependent/independent versus local/global/loop are different classifications.

Question sources

Compiler Design review questions · Question 9

Question 157 | NS0237

What does 'register spilling' mean in the context of register allocation?

A. Allocating the same register to two variables |

B. Storing a variable in memory instead of a register because no register is available

C. Using floating-point registers for integer values |

D. Eliminating unnecessary register copies

Hint

Match stored information to its lifetime, access path and calling-convention role; distinguish compile-time organization from run-time storage.

Answer B: Store selected values in memory and insert loads/stores when registers do not suffice.

COMP36 says spilling places selected values in memory when the register assignment cannot fit available registers. Loads and stores then connect memory-resident values to instructions needing registers. Register sharing can be legal for nonoverlapping live ranges, while copy elimination and choosing a different register class are separate operations rather than this storage fallback.

Why each choice is right or wrong

A - Incorrect: Register sharing may be legal for disjoint live ranges and does not itself describe spilling.

B - Correct: Spilling moves selected values to memory because a feasible register-only assignment is unavailable.

C - Incorrect: Using another register class is not the definition of writing spilled values to memory.

D - Incorrect: Removing redundant register copies is a different optimization from memory-backed spilling.

Primary note proof

Study-note section COMP36

> A value is live where it can be used before redefinition. In the interference model, values that must coexist in distinct registers are connected by an edge; each register is a color. Nonoverlapping live ranges may share a register. If a valid assignment cannot fit the available registers, spilling places selected values in memory and inserts loads/stores.

Study-note section COMP36

The general graph k-colorability decision problem is NP-complete for fixed k≥3; optimal register assignment/spill minimization is a hard optimization problem in general, so practical allocators use heuristics.

Question sources

Compiler Design review questions · Question 10

Question 158 | NS0238

For Java numeric conversions such as int→long and int→float, which statement correctly describes widening?

A. Converting a wider type to a narrower type, possibly losing information |

B. Converting to a numeric type with a wider representable range; some widening conversions can still lose precision.

C. Declaring a variable with an explicit type annotation |

D. Assigning a value to a pointer of a different type

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer B: Converting to a numeric type with a wider representable range; some widening conversions can still lose precision.

Java int→long is widening and exact, while int→float is also widening even though some large integer values round. Widening therefore does not universally guarantee preservation of every bit of precision. Narrowing goes toward a more restricted representation; type annotations and arbitrary pointer changes are separate concepts. Option B captures the relevant larger-range conversion without making the false no-loss promise.

Why each choice is right or wrong

A - Incorrect: This describes narrowing rather than widening.

B - Correct: Widening can increase range while still losing precision, as int→float illustrates.

C - Incorrect: An explicit annotation declares a type rather than converting a value.

D - Incorrect: An arbitrary pointer-type change does not define Java numeric widening.

Primary note proof

Study-note section COMP23

Coercion is an implicit conversion permitted by the language; a cast requests an explicit conversion. Widening often increases representable range, but can still lose precision: Java int→float is widening although large integers round. Java int→long is exact. Narrowing can lose range or fractional information.

Study-note section COMP23

> In C++ ordinary int x=3.14 converts to 3 when representable; it is not inherently a type error. int x{3.14} is rejected as narrowing. C++ floating-to-integer conversion discards the fraction and has undefined behavior if the truncated value is unrepresentable. Java ordinary double→int assignment needs a permitted explicit cast. Mixing char and int is ordinarily valid after promotion in C++/Java; do not import a fictitious universal ban.

Question sources

Compiler Design review questions · Question 11

Question 159 | NS0239

Which of the following data structures is most commonly used to implement a symbol table for fast insertion and lookup?

A. Sorted linked list |

B. Hash table

C. Binary search tree |

D. Stack

Hint

Ask whether the required information is a declaration’s identity and attributes, program structure or the ordered source tokens.

Answer B: A hash table.

COMP19 directly identifies hashing as giving expected constant-time insertion and lookup under suitable hashing. A stack supports nested scope order but is not the primary keyed mapping. Lists and trees are possible alternatives, yet neither is the specific common expected-constant-time structure emphasized in the note and requested by the stem.

Why each choice is right or wrong

A - Incorrect: A sorted linked list generally requires linear search and does not supply the highlighted expected constant lookup.

B - Correct: Hash tables commonly implement expected O(1) symbol insertion and lookup.

C - Incorrect: A search tree is possible, but the note specifically identifies hashing for the usual fast keyed implementation.

D - Incorrect: A stack organizes scope nesting rather than providing fast arbitrary name lookup within a scope.

Primary note proof

Study-note section COMP19

> A symbol table maps names or symbol identities to attributes: type, scope, width, storage offset, parameter signature, constant information or linkage. A hash table gives expected O(1)

insertion/lookup with suitable hashing; a stack of scope tables supports lexical nesting. Search the nearest scope first, insert declarations in the current scope, and remove visibility when leaving it.

Study-note section COMP19

Scanning may intern a spelling first; semantic analysis determines the declaration and type. Implementations differ, so first interaction is conventional rather than universal. Not every punctuation token needs a symbol-table entry; literals may use a separate literal table. A hash table is an implementation, not the information’s semantic role.

Question sources

Compiler Design review questions · Question 12

Question 160 | NS0240

A grammar is said to be 'left-factored' when:

A. All productions are left-recursive |

B. Productions for the same non-terminal that share a common prefix are rewritten to factor out the prefix

C. The grammar has no ε-productions |

D. The grammar can be parsed by a bottom-up parser only

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: Common prefixes are factored out of alternatives.

COMP12 rewrites A→αβ|αγ as A→αA′ and A′→β|γ. The transformation postpones the alternative choice until after consuming the common prefix. It does not mean making productions left-recursive, removing all epsilon rules or restricting parsing to bottom-up methods; left factoring alone also does not guarantee LL(1).

Why each choice is right or wrong

A - Incorrect: Left recursion and left factoring are distinct properties and transformations.

B - Correct: Factoring extracts shared prefixes and delays selection between the remaining suffix alternatives.

C - Incorrect: Factoring does not require removing every epsilon production.

D - Incorrect: Factored grammars can support top-down prediction; bottom-up exclusivity is not a defining condition.

Primary note proof

Study-note section COMP12

> Left factoring rewrites A→αβ|αγ using:
> A→αA′
> A′→β|γ
> It postpones a choice until after the common prefix. It does not by itself remove ambiguity or guarantee LL(1). LR parsers can handle suitable left-recursive grammars; ordinary unmodified recursive descent can recurse forever before consuming input.

Question sources

Compiler Design review questions · Question 13

Question 161 | NS0241

What is 'constant propagation' in compiler optimization?

A. Evaluating constant expressions at compile time |

B. Replacing uses of a variable with its known constant value at each program point

C. Removing code that has no effect on program output |

D. Moving constants from memory to registers

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer B: Replace uses with their known constant values.

COMP30 gives x=3; y=x+2 becoming y=3+2 before folding. Replacing x by its known value is propagation; evaluating 3+2 is folding. Removing unnecessary code or choosing register locations addresses different transformations. The program-point qualification matters because the value must still be known on all relevant incoming paths.

Why each choice is right or wrong

A - Incorrect: Evaluating an already constant expression is constant folding, which can follow propagation.

B - Correct: Propagation substitutes a variable’s established constant value at uses where that fact holds.

C - Incorrect: Removing effect-free unused or unreachable work is dead-code elimination.

D - Incorrect: Memory-to-register placement is not the substitution of a known compile-time value.

Primary note proof

Study-note section COMP30

> Constant folding evaluates constant expressions during compilation: 3*4→12. Constant propagation substitutes known values: x=3; y=x+2 becomes y=3+2 before folding. Copy propagation substitutes a copied name while its source remains unchanged. Dead-code elimination removes unreachable code or computations whose values are unused and whose evaluation has no necessary side effects.

Question sources

Compiler Design review questions · Question 14

Question 162 | NS0242

In an LR parsing table, what does a 'reduce/reduce conflict' indicate?

A. Two different productions can both be used to reduce the same stack contents with the same lookahead |

B. The parser must both shift and reduce on the same input symbol

C. The grammar has a dangling else problem |

D. The grammar contains ε-productions

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer A: Two competing reductions for the same parser state/lookahead.

COMP17 defines reduce/reduce as two reductions being available. In a parsing table that means the same state and lookahead permit different completed productions. A shift competing with reduction is another conflict type. Dangling else is a grammar ambiguity example, and epsilon productions alone do not guarantee any reduce/reduce collision.

Why each choice is right or wrong

A - Correct: The table offers two different production reductions for the same state and lookahead.

B - Incorrect: Simultaneous shift and reduce actions define shift/reduce rather than reduce/reduce.

C - Incorrect: Dangling else commonly creates an attachment ambiguity, not the definition of two competing reductions.

D - Incorrect: Epsilon rules can appear in conflict-free grammars and do not alone imply this collision.

Primary note proof

Study-note section COMP17

> SLR(1) uses LR(0) states and FOLLOW(A) for reduction by a completed A production. Canonical LR(1) attaches context-specific lookahead to items. LALR(1) merges canonical states with the same LR(0) core, combining lookaheads; merging can introduce reduce/reduce conflicts. The standard grammar-class hierarchy is LR(0)⊆SLR(1)⊆LALR(1)⊆LR(1), with increasing capability and often table size.

Study-note section COMP17

> A shift/reduce conflict offers both consuming a token and reducing; reduce/reduce offers two reductions. An SLR or LALR conflict does not by itself prove the grammar ambiguous. Canonical LR(1) handles LR(1) grammars, not every unambiguous CFG. Precedence and associativity can resolve suitable expression conflicts; changing parser names alone does not repair all ambiguity.

Question sources

Compiler Design review questions · Question 15

Question 163 | NS0243

Which compiler phase is primarily responsible for detecting the incompatible assignment in the Java declaration int x = "hello";?

A. Lexical analysis |

B. Syntax analysis

C. Semantic analysis |

D. Code generation

Hint

Check operand compatibility and permitted conversions under the stated language rules; distinguish grammatical structure from semantic meaning.

Answer C: Semantic analysis |

The declaration has recognizable tokens and a valid declaration-shaped syntax, but assigns a string to an integer variable. Under Java's type rules this violates assignment compatibility, so semantic analysis and type checking identify the error. Scanning recognizes the identifier, type keyword and literal; parsing organizes them. Code generation uses the checked representation later. The named language removes any assumption that every statically typed language has identical implicit conversions.

Why each choice is right or wrong

A - Incorrect: The individual tokens are recognizable; the error is their type compatibility.

B - Incorrect: The declaration's grammar structure is not the primary problem.

C - Correct: Semantic type checking rejects the string-to-integer assignment.

D - Incorrect: Code generation follows the frontend type check rather than defining this compatibility rule.

Primary note proof

Study-note section COMP08

> A lexical error prevents forming a legal token, such as an invalid character or unterminated literal. A syntax error is an illegal token arrangement, such as a missing delimiter. A semantic error violates name, scope or type rules after grammatical recognition. A wrong algorithm can compile successfully; compilers do not detect all logical mistakes.

Study-note section COMP22

> int x="hello" is a type mismatch in ordinary C++/Java; division by zero and bounds violations are not automatically operand-type mismatches.

Question sources

Compiler Design review questions · Question 16

Question 164 | NS0244

In Thompson's construction for building an NFA from a regular expression, the NFA for the concatenation r1r2 is built by:

A. Merging the start states of both NFAs |

B. Adding an ε-transition from the accepting state(s) of the NFA for r1 to the start state of the NFA for r2

C. Creating a new start state with ε-transitions to both NFAs |

D. Taking the cross product of the two NFAs

Hint

Follow the representation at each construction step and apply the pattern or state-set rules consistently, including empty cases.

Answer B: Connect the first fragment’s accepting end to the second start with ε.

COMP06 directly gives the concatenation rule: connect r’s final state to s’s start by epsilon. This forces any complete path through the first fragment to continue through the second in order. Branching epsilon paths instead implement alternation; merged starts and product constructions do not implement the specified Thompson concatenation step.

Why each choice is right or wrong

A - Incorrect: Merging starts does not force a complete r1 match followed by an r2 match.

B - Correct: The epsilon connection joins the first accepting end to the second start without consuming input.

C - Incorrect: A new branching start with epsilon alternatives implements union rather than concatenation.

D - Incorrect: A state cross product is not the Thompson fragment connection for concatenation.

Primary note proof

Study-note section COMP06

> A common generator pipeline is regular expressions → NFA using Thompson construction → DFA using subset construction; minimization is optional. For concatenation rs, connect r’s final state to s’s start by ε. Alternation introduces branching ε paths; star adds a way to skip and repeat the fragment.

Study-note section COMP06

> The ε-closure of states includes those reachable by zero or more ε edges. A DFA state is a set of NFA states; its a-transition is ε-closure(move(set,a)). Start with the NFA start closure, repeatedly add unseen subsets, and mark subsets containing an NFA final state as accepting. Token identity and rule priority annotate accepting states; the empty subset acts as a dead state.

Question sources

Compiler Design review questions · Question 17

Question 165 | NS0245

What is the 'dangling else' problem?

A. An else branch that can never be reached due to preceding conditions |

B. Ambiguity in grammars that allows an else to be matched with either of two if statements

C. An else clause that lacks a body |

D. An else used outside an if statement

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Ambiguous attachment of else to one of two nested ifs.

COMP10 describes an else that can attach to either nested if and resolves it with nearest-unmatched-if policy or a grammar rewrite. That grammatical ambiguity is the dangling-else problem. It does not mean unreachable code, a missing body or an else outside any if, which are different semantic or syntactic situations.

Why each choice is right or wrong

A - Incorrect: Unreachability concerns control-flow feasibility, not the competing grammatical attachment of else.

B - Correct: The ambiguity is whether else belongs to the inner or outer unmatched if.

C - Incorrect: A missing else body is a syntax defect rather than the classic attachment ambiguity.

D - Incorrect: An else outside any if is not the two-attachment situation defining dangling else.

Primary note proof

Study-note section COMP10

> The dangling-else grammar permits else to attach to either nested if; a nearest-unmatched-if rule or grammar rewrite resolves it. Taking a leftmost derivation by itself does not resolve ambiguous interpretation.

Question sources

Compiler Design review questions · Question 18

Question 166 | NS0246

Which representation explicitly uses basic blocks as vertices and possible transfers as edges, making it the direct structural basis for dominator and natural-loop analysis?

A. Abstract Syntax Tree (AST) |

B. Three-Address Code (TAC)

C. Static Single Assignment (SSA) form |

D. Control Flow Graph (CFG)

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer D: Control Flow Graph (CFG)

A control-flow graph explicitly records basic blocks and their possible successor transfers. Dominator relationships describe which blocks occur on every path to another block; back edges to dominating headers identify natural loops. TAC can populate the blocks, and SSA can coexist with this graph, but neither alone names the requested vertex/edge structure. An AST emphasizes source syntax rather than this control-flow representation.

Why each choice is right or wrong

A - Incorrect: An AST represents source hierarchy rather than the specified block-transfer graph.

B - Incorrect: TAC gives instructions, not by itself the requested graph.

C - Incorrect: SSA is a definition-naming discipline that can coexist with a CFG.

D - Correct: A CFG explicitly provides the specified blocks, edges and path structure.

Primary note proof

Study-note section COMP28

> A basic block is a maximal straight-line instruction sequence entered at its beginning and exited at its end. Leaders are the first instruction, every jump target, and each instruction after a branch. Start a block at each leader and continue until the next leader.

Study-note section COMP28

> A control-flow graph has blocks as vertices and possible next-block transfers as edges, including fall-through. A node D dominates N when every entry-to-N path passes through D. For an edge N→H where H dominates N, H is a loop header of a natural loop. This structure supports loop analysis. An expression DAG records value dependencies, not arbitrary control flow.

Question sources

Compiler Design review questions · Question 19

Question 167 | NS0247

In Static Single Assignment (SSA) form, what is the purpose of a 'φ (phi) function'?

A. To represent a conditional branch in the control flow graph |

B. To select the correct version of a variable at join points where multiple control flow paths merge

C. To mark the entry point of a function |

D. To represent a loop-back edge in the CFG

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer B: Select the incoming variable version corresponding to the predecessor taken.

COMP27 defines phi at a join as choosing an incoming value according to the actual predecessor path. It is not a conditional branch instruction, a function-entry marker or a back edge, and does not average its operands. The function reconciles distinct definition names when different paths reach the same use.

Why each choice is right or wrong

A - Incorrect: A phi function selects an incoming value; it does not transfer control between blocks.

B - Correct: It merges definition versions by selecting the value associated with the predecessor actually taken.

C - Incorrect: Function entry is a control-flow location, not the value-selection purpose of phi.

D - Incorrect: A back edge represents loop control transfer rather than a merge of SSA values.

Primary note proof

Study-note section COMP27

> A conditional expression can generate if a<b goto Ltrue; goto Lfalse; labels join control paths. Short-circuit AND evaluates the right operand only after a true left operand; OR only after a false one. Translating both eagerly can change side effects. For a zero-based array with element width w and lower bound L, element i uses base+(i−L)*w, assuming fixed-width contiguous storage.

Study-note section COMP27

> SSA gives each definition a distinct name. At a join, x3=φ(x1 from B1,x2 from B2) chooses the incoming value according to the predecessor actually taken; it is not an arithmetic average. SSA is a naming discipline over IR and commonly coexists with a control-flow graph.

Question sources

Compiler Design review questions · Question 20

Question 168 | NS0248

The FOLLOW set of the start symbol S always contains:

A. ε (epsilon) |

B. The end-of-input marker $

C. The first terminal of every production |

D. Nothing; FOLLOW(S) is always empty

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: The end-of-input marker $.

COMP13 explicitly places $ in FOLLOW(S), representing that a completed start-symbol derivation can be followed by end of input. FOLLOW never contains epsilon. Possible initial terminals belong to FIRST, not automatically FOLLOW. Therefore the marker is guaranteed even when other following terminals depend on the grammar.

Why each choice is right or wrong

A - Incorrect: FOLLOW contains terminals and possibly the end marker, never epsilon.

B - Correct: The end-of-input marker is placed in FOLLOW of the start symbol by initialization.

C - Incorrect: Initial terminals are FIRST information and are not automatically following terminals.

D - Incorrect: FOLLOW(S) is not empty because it contains the end marker.

Primary note proof

Study-note section COMP13

> FIRST(α) contains terminals that can begin strings derived from α, plus ε when α can vanish. For a sequence, add FIRST of each symbol except ε, continue while symbols are nullable, and add ε only if every symbol is nullable.

Study-note section COMP13

> FOLLOW(A) contains possible next terminals after A, plus end marker $ where appropriate; never ε. Put $ in FOLLOW(S). For B→αAβ, add FIRST(β)−{ε} to FOLLOW(A); if β is nullable, add FOLLOW(B). Repeat all rules until no sets grow.

Question sources

Compiler Design review questions · Question 21

Question 169 | NS0249

For an unsigned fixed-width integer x with width greater than three and modular arithmetic, which optimization replaces x = x * 8 with x = x << 3?

A. Constant folding |

B. Dead code elimination

C. Strength reduction |

D. Loop unrolling

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer C: Strength reduction |

The transformation replaces multiplication by a power of two with an equivalent shift under the stated unsigned model. This is strength reduction when the target makes that replacement cheaper. It does not evaluate a constant-only expression because x remains variable, does not remove an unused statement, and does not replicate loop iterations. The explicit arithmetic model matters because signed-language overflow or shift rules can make a superficially similar rewrite invalid.

Why each choice is right or wrong

A - Incorrect: The operand x remains variable, so this is not evaluation of an all-constant expression.

B - Incorrect: The assignment remains and still computes a needed result.

C - Correct: Replacing an operation by an equivalent cheaper form is strength reduction.

D - Incorrect: No loop body is replicated.

Primary note proof

Study-note section COMP31

> Algebraic simplification removes identities when valid: integer x+0→x. Strength reduction replaces an operation with an equivalent cheaper form, such as unsigned x*8→x<<3 for a valid width and shift count. Target costs determine whether it is actually cheaper.

Study-note section COMP33

> Signed C/C++ overflow and shift rules differ from unsigned modular arithmetic; a mathematical identity is not automatically a legal transformation.

Study-note section COMP32

> Unrolling replicates a body to reduce branch overhead and expose parallelism; handle remaining iterations and increased code size.

Question sources

Compiler Design review questions · Question 22

Question 170 | NS0250

In a compiler's code generator, 'calling convention' refers to:

A. The order in which source files are compiled |

B. The agreed-upon protocol for how functions pass arguments, return values, and manage the stack

C. The naming convention for generated labels |

D. The format of debug information in the object file

Hint

Match stored information to its lifetime, access path and calling-convention role; distinguish compile-time organization from run-time storage.

Answer B: The protocol for arguments, results, registers and stack management.

COMP37 defines the locations of arguments/results, saved-register responsibilities, stack alignment and frame duties as the calling convention. These permit caller and callee to agree on the runtime call interface. Source-file build order, label spelling and debug format concern other compiler/toolchain conventions rather than this execution protocol.

Why each choice is right or wrong

A - Incorrect: Compilation order concerns build organization rather than the runtime interface between callers and callees.

B - Correct: The calling convention specifies argument/result passing and register/stack responsibilities.

C - Incorrect: Label naming is separate from the runtime storage and control-transfer call protocol.

D - Incorrect: Debug information format is not the protocol used to invoke and return from functions.

Primary note proof

Study-note section COMP37

> Static storage lasts throughout execution; stack frames support nested calls and recursion; heap storage supports lifetimes beyond a call. A frame may contain parameters, return address, saved registers, locals and temporaries. Sizes, alignment and offsets are target-dependent; a two-byte int is an example, not a C++ guarantee.

Study-note section COMP37

> A calling convention specifies argument and result locations, caller/callee-saved registers, stack alignment and frame responsibilities. Escape analysis asks whether an object/reference can

outlive or be accessed outside a region such as its allocating call. Proven nonescaping allocations may use stack storage or be eliminated by scalar replacement, provided identity and lifetime semantics remain correct.

Question sources

Compiler Design review questions · Question 23

Question 171 | NS0251

What does 'peephole optimization' involve?

A. Optimizing entire functions at once |

B. Examining and replacing small sequences of generated instructions with more efficient equivalents

C. Inlining small functions at call sites |

D. Eliminating unreachable basic blocks

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer B: Examine short instruction sequences and replace them with better equivalents.

COMP31 defines peephole optimization by short instruction windows and lists redundant-load removal, machine idioms and jump-chain shortening. B states that method directly. Whole-function optimization has a broader scope; inlining changes calls to bodies, and unreachable-block removal is a particular control-flow simplification rather than the definition of examining small windows.

Why each choice is right or wrong

A - Incorrect: Whole-function scope is broader than the short instruction windows defining peephole optimization.

B - Correct: The optimizer inspects short instruction sequences and substitutes equivalent more efficient forms.

C - Incorrect: Inlining expands calls into bodies rather than defining the local instruction-window method.

D - Incorrect: Unreachable-block removal is one control-flow optimization, not the defining peephole inspection mechanism.

Primary note proof

Study-note section COMP31

> Peephole optimization examines short instruction windows. It can remove redundant loads/stores, replace machine idioms and shorten jump chains: jump L1 where L1 immediately jumps L2 can become jump L2.

Question sources

Compiler Design review questions · Question 24

Question 172 | NS0252

Which statement correctly compares LALR(1) with SLR(1) and canonical LR(1), allowing the case where no equal-core canonical states need merging?

A. LALR(1) is weaker than SLR(1) |

B. LALR(1) has no more states than canonical LR(1) and can resolve some conflicts SLR(1) cannot.

C. LALR(1) and SLR(1) produce identical parsing tables for all grammars |

D. LALR(1) requires more lookahead than SLR(1)

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: LALR(1) has no more states than canonical LR(1) and can resolve some conflicts SLR(1) cannot.

LALR combines lookaheads of canonical LR(1) states with equal LR(0) cores. This gives no more states than the canonical machine, with a reduction when distinct equal-core states merge. LALR can handle some grammars beyond SLR because its lookaheads are more precise. Both use one lookahead token, and their tables are not identical for every grammar.

Why each choice is right or wrong

A - Incorrect: The hierarchy places SLR within LALR rather than the reverse.

B - Correct: This states both the correct state-count bound and the greater parsing capability.

C - Incorrect: The methods can use different reduction lookaheads and tables.

D - Incorrect: Both use one lookahead token.

Primary note proof

Study-note section COMP17

> SLR(1) uses LR(0) states and FOLLOW(A) for reduction by a completed A production. Canonical LR(1) attaches context-specific lookahead to items. LALR(1) merges canonical states with the same LR(0) core, combining lookaheads; merging can introduce reduce/reduce conflicts. The standard grammar-class hierarchy is LR(0)⊆SLR(1)⊆LALR(1)⊆LR(1), with increasing capability and often table size.

Study-note section COMP17

> A shift/reduce conflict offers both consuming a token and reducing; reduce/reduce offers two reductions. An SLR or LALR conflict does not by itself prove the grammar ambiguous. Canonical LR(1) handles LR(1) grammars, not every unambiguous CFG. Precedence and associativity can resolve suitable expression conflicts; changing parser names alone does not repair all ambiguity.

Question sources

Compiler Design review questions · Question 25

Question 173 | NS0253

In data flow analysis, 'live variable analysis' is a:

A. Forward may-analysis |

B. Backward may-analysis

C. Forward must-analysis |

D. Backward must-analysis

Hint

Track which facts can hold along one path and which must hold along every path. Apply the specified direction, transfer rule and join operation.

Answer B: Backward may-analysis.

COMP29 explicitly calls liveness backward may-analysis. A value is live if some future path reads it before redefinition, so information flows backward from successor uses and combines possible uses rather than requiring every path. This distinguishes both the direction and may-versus-must dimensions requested by the options.

Why each choice is right or wrong

A - Incorrect: May is appropriate, but liveness propagates from future uses backward rather than forward.

B - Correct: Liveness asks whether some future path uses a value before redefinition, a backward may property.

C - Incorrect: Both direction and must quantification differ from the liveness definition.

D - Incorrect: Backward direction fits, but liveness requires a possible use, not use on every path.

Primary note proof

Study-note section COMP29

> Liveness is backward may-analysis: a value might be read before redefinition on some future path. OUT[B]=union of IN over successors; IN[B]=USE[B]∪(OUT[B]−DEF[B]). Iterate to a fixed point. Liveness supports register allocation and dead-value removal. Definite-assignment analysis asks whether initialization occurs on every path before use; reaching definitions alone does not prove this.

Question sources

Compiler Design review questions · Question 26

Question 174 | NS0254

The 'closure' operation in NFA-to-DFA subset construction computes:

A. The set of all NFA states reachable from a given set of states using only ε-transitions |

B. The set of all NFA states reachable using any number of any transitions

C. The minimal DFA equivalent to the NFA |

D. The FIRST set of a non-terminal

Hint

Follow the representation at each construction step and apply the pattern or state-set rules consistently, including empty cases.

Answer A: States reachable by zero or more ε transitions.

COMP06 defines epsilon closure as all states reachable without consuming input through zero or more epsilon edges. Zero moves includes the starting set itself. Arbitrary labeled transitions would consume symbols and belong to a different reachability relation; minimization and FIRST-set computation are separate algorithms. A therefore matches the closure used in subset construction.

Why each choice is right or wrong

A - Correct: Epsilon closure includes the starting set and every state reachable using only epsilon edges.

B - Incorrect: Arbitrary symbol transitions consume input and are not part of epsilon closure.

C - Incorrect: Closure supplies subsets during determinization; it does not itself minimize the resulting DFA.

D - Incorrect: FIRST is a grammar lookahead set, not epsilon reachability among NFA states.

Primary note proof

Study-note section COMP06

> A common generator pipeline is regular expressions → NFA using Thompson construction → DFA using subset construction; minimization is optional. For concatenation rs, connect r's final state to s's start by ε. Alternation introduces branching ε paths; star adds a way to skip and repeat the fragment.

Study-note section COMP06

> The ε-closure of states includes those reachable by zero or more ε edges. A DFA state is a set of NFA states; its a-transition is ε-closure(move(set,a)). Start with the NFA start closure, repeatedly add unseen subsets, and mark subsets containing an NFA final state as accepting. Token identity and rule priority annotate accepting states; the empty subset acts as a dead state.

Question sources

Compiler Design review questions · Question 27

Question 175 | NS0255

For a real-machine backend, which listed optimization chiefly depends on the target's instruction latencies and execution resources?

A. Constant folding |

B. Dead code elimination

C. Instruction scheduling |

D. Loop-invariant code motion

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer C: Instruction scheduling |

Instruction scheduling chooses a legal instruction order while considering target resources and latencies. It is therefore the target-dependent backend choice in this comparison. Constant folding computes known expressions, dead-code elimination removes unnecessary computation while preserving effects, and loop-invariant motion relocates safe unchanged expressions. Their validity depends on program semantics, but these three need not be organized around a particular processor's scheduling resources.

Why each choice is right or wrong

A - Incorrect: Folding evaluates known expressions from program semantics rather than scheduling machine resources.

B - Incorrect: Dead-code elimination uses reachability, use and side-effect information.

C - Correct: Instruction scheduling considers target execution resources and instruction latencies.

D - Incorrect: Invariant motion uses loop values, paths and safety conditions rather than target issue scheduling.

Primary note proof

Study-note section COMP-S04

> Instruction scheduling orders selected target instructions while preserving dependencies. A target-aware scheduler also considers instruction latencies and the processor resources needed by each operation. This is a backend optimization whose performance choices depend on the target machine, even when a reusable scheduling framework serves many targets.

Study-note section COMP-S04

> By contrast, constant folding, dead-code elimination and safe loop-invariant motion can be expressed on a target-neutral IR using values, control flow and side-effect rules. Target costs can still influence whether a legal transformation is worthwhile. Machine independence therefore describes the abstraction of a pass, not freedom from all language semantics or implementation costs.

Question sources

Compiler Design review questions · Question 28

Question 176 | NS0256

In a context-free grammar, a 'sentential form' is:

A. A string consisting only of terminal symbols |

B. Any string of terminals and non-terminals derivable from the start symbol

C. A string that cannot be further reduced |

D. The right-hand side of a production rule

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Any terminal/nonterminal string derivable from the start symbol.

COMP09 defines a sentential form as any string of terminals and nonterminals derivable from S. Terminal-only sentences are a special case, not the full definition. A production right side need not itself be reachable from S, and irreducibility does not define derivability. B captures the required reachability and permits both kinds of symbols.

Why each choice is right or wrong

A - Incorrect: Terminal-only strings are sentences, a subset of sentential forms rather than their full definition.

B - Correct: Derivability from the start symbol with terminals and possibly nonterminals is the defining property.

C - Incorrect: A sentential form can still contain expandable nonterminals; inability to reduce is not required.

D - Incorrect: A right-hand side is not necessarily reachable from the grammar's start symbol.

Primary note proof

Study-note section COMP09

> A context-free grammar G=(N,T,P,S) has nonterminals N, terminals T, productions P and start S. A production A→α rewrites one nonterminal. A sentential form is any string of terminals/nonterminals derivable from S; a sentence contains only terminals. A leftmost derivation rewrites the leftmost remaining nonterminal; a rightmost one rewrites the rightmost.

Study-note section COMP09

> A parse tree shows grammar productions: root S, internal nonterminals, and terminal leaves, with ε sometimes shown for empty productions. Its left-to-right yield is the input. An AST omits grammar scaffolding and punctuation; expression operands are leaves and operators internal nodes. The parser can build either representation, depending on implementation.

Question sources

Compiler Design review questions · Question 29

Question 177 | NS0257

What is 'function inlining' and what is its primary risk?

A. Replacing a function call with the function body; risk is incorrect semantics |

B. Replacing a function call with the function body; risk is increased code size (code bloat)

C. Moving a function to a different module; risk is linker errors |

D. Specializing a function for specific argument values; risk is type errors

Hint

Match stored information to its lifetime, access path and calling-convention role; distinguish compile-time organization from run-time storage.

Answer B: Replace a call with its body; excessive code size is a risk.

COMP38 defines inlining as substituting a suitably renamed body while preserving argument evaluation, returns and side effects. It can remove call overhead but increase code size and instruction-cache pressure. Correct inlining preserves semantics; corruption is an implementation bug rather than its defining trade-off. Moving modules or specializing arguments describes different transformations.

Why each choice is right or wrong

A - Incorrect: Inlining must preserve semantics; incorrect behavior is a bug rather than the normal primary trade-off stated here.

B - Correct: Body substitution can reduce call overhead while increasing generated code size.

C - Incorrect: Moving a function between modules is not call-site body substitution.

D - Incorrect: Argument specialization is a different optimization and does not define general function inlining.

Primary note proof

Study-note section COMP38

> Inlining replaces a call with a suitably renamed copy of its body, preserving argument evaluation, returns and side effects. It can remove call overhead and expose optimization opportunities, but may increase code size and instruction-cache pressure. A recursive function cannot simply be expanded without limit.

Study-note section COMP38

> Virtual dispatch chooses an overridden method based on a receiver’s runtime class. A common implementation uses a per-class virtual-method table and an object pointer to the table; the call loads the selected entry. This is an implementation strategy, not a universal language requirement. Name mangling distinguishes linker symbols, often for overloads; it is not itself runtime method selection. Proven receiver types may permit devirtualization.

Question sources

Compiler Design review questions · Question 30

Question 178 | NS0258

Which of the following best describes the 'handle' in bottom-up parsing?

A. The leftmost terminal in the input string |

B. A substring of the current sentential form that matches the right-hand side of a production and whose reduction leads to a valid parse

C. The top element of the parsing stack |

D. The lookahead token used for parsing decisions

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: The RHS occurrence whose next reverse-rightmost reduction leads toward the valid parse.

COMP15 defines a handle as a production RHS occurrence whose reduction is the next reverse-rightmost derivation step. B is the closest complete description: its valid-parse condition must mean that next legitimate step, not merely any matching substring. Neither a single stack item nor the lookahead alone specifies that contextual production occurrence.

Why each choice is right or wrong

A - Incorrect: The leftmost terminal need not be the handle and does not identify a production RHS occurrence.

B - Correct: Read valid reduction as the next reverse-rightmost step; contextual validity distinguishes a handle from arbitrary matches.

C - Incorrect: A handle can span multiple symbols and is not simply the top stack element.

D - Incorrect: Lookahead guides action selection but is not the substring being reduced.

Primary note proof

Study-note section COMP15

> Bottom-up parsing starts from input and reduces toward S, constructing a rightmost derivation in reverse. A handle is a production RHS occurrence whose reduction is the next step of that reverse derivation. Not every substring matching a RHS is a handle in the current context.

Study-note section COMP15

> Shift pushes the next input token and a state; reduce replaces a recognized RHS by its LHS and consults GOTO; accept recognizes completion; error reports an invalid situation. For A→XYZ, pop three grammar symbols and their states, expose state q, then push A and GOTO[q,A]. “Go” is not a separate standard shift-reduce ACTION, although GOTO is essential table machinery.

Question sources

Compiler Design review questions · Question 31

Question 179 | NS0259

In the context of code generation, what is a 'basic block'?

A. A block of memory allocated on the heap |

B. A maximal straight-line sequence of instructions with one entry point and one exit point

C. A function or procedure in the source code |

D. A group of related variable declarations

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer B: A maximal straight-line sequence with entry at the beginning and exit at the end.

COMP28 directly defines this sequence as a basic block. Maximality means it is not arbitrarily shortened while retaining the same entry/exit structure. Heap allocation, whole functions and related declarations describe different entities; a function may contain many basic blocks separated by branch targets and control transfers.

Why each choice is right or wrong

A - Incorrect: A heap memory region is storage, not a straight-line control-flow unit.

B - Correct: This matches the maximal straight-line single-entry, end-exit definition in COMP28.

C - Incorrect: A function may contain branches and multiple basic blocks rather than one straight-line block.

D - Incorrect: Related declarations are not the instruction sequence defined by control-flow boundaries.

Primary note proof

Study-note section COMP28

> A basic block is a maximal straight-line instruction sequence entered at its beginning and exited at its end. Leaders are the first instruction, every jump target, and each instruction after a branch. Start a block at each leader and continue until the next leader.

Study-note section COMP28

> A control-flow graph has blocks as vertices and possible next-block transfers as edges, including fall-through. A node D dominates N when every entry-to-N path passes through D. For an edge N→H where H dominates N, H is a loop header of a natural loop. This structure supports loop analysis. An expression DAG records value dependencies, not arbitrary control flow.

Question sources

Compiler Design review questions · Question 32

Question 180 | NS0260

What is 'alias analysis' and why is it important for optimization?

A. Determining which variables have the same name in different scopes; important for scope resolution |

B. Determining whether two pointer expressions can refer to the same memory location; important because aliasing restricts reordering and CSE

C. Detecting variables with the same type; important for type coercion |

D. Finding duplicate function names; important for linking

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer B: Determine whether references may denote the same memory, constraining transformations.

COMP33 asks whether references can designate the same memory. A write through p can invalidate a value loaded through q unless nonaliasing or another proof permits reuse. Therefore alias facts constrain CSE and reordering. Identical spellings, types or function names concern scope, typing or linkage rather than memory-location overlap.

Why each choice is right or wrong

A - Incorrect: Equal spelling in different scopes concerns declaration identity rather than whether references share memory.

B - Correct: Potential shared locations determine whether writes can invalidate reused loads or reordered memory operations.

C - Incorrect: Equal types do not prove shared addresses and are not the alias-analysis question.

D - Incorrect: Duplicate linker names concern symbol resolution rather than potential overlap of referenced memory.

Primary note proof

Study-note section COMP33

> Optimization preserves defined observable behavior, including required side effects and exceptions. Alias analysis asks whether references can denote the same memory. After *p=5, a previous load through q cannot be reused unless p and q are known not to alias or an equivalent proof applies. Function calls can also invalidate memory facts.

Study-note section COMP33

> Do not move a potentially trapping division outside a loop that might execute zero times. Floating-point reassociation can change rounding and NaN behavior. Signed C/C++ overflow and shift rules differ from unsigned modular arithmetic; a mathematical identity is not automatically a legal transformation. Dependence analysis must show that fusion or scheduling preserves the required order of memory accesses.

Question sources

Compiler Design review questions · Question 33

Question 181 | NS0261

Which statement about recursive-descent parsers is TRUE?

A. They cannot handle left-recursive grammars without modification |

B. They inherently support all context-free grammars

C. They require a parsing table to function |

D. They build the parse tree bottom-up

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer A: Ordinary recursive descent needs modification to handle left recursion.

COMP12 explains that ordinary unmodified recursive descent can recurse forever on left recursion before consuming a token. Removing left recursion or changing the algorithm is therefore necessary for that direct implementation. Recursive descent is top-down and normally procedural, so it does not inherently require an explicit table or support every CFG unchanged.

Why each choice is right or wrong

A - Correct: Without transformation or algorithmic modification, left recursion can cause infinite recursive calls before input consumption.

B - Incorrect: Ordinary recursive descent is not an unrestricted parser for every context-free grammar.

C - Incorrect: Procedures can encode production choices directly; an explicit parsing table is not required.

D - Incorrect: Recursive descent expands productions top-down rather than constructing a reverse-rightmost bottom-up parse.

Primary note proof

Study-note section COMP12

> Immediate left recursion has $A \to A\alpha$. For $A \to A\alpha \mid \beta$, where β does not start with A, introduce a new nonterminal A' and use:
> $A \to \beta A'$
> $A' \to \alpha A' \mid \varepsilon$
> This preserves the generated language; tree shape changes, so semantic actions must preserve intended associativity. Indirect left recursion follows a cycle through other nonterminals and also needs removal for ordinary predictive descent.

Study-note section COMP12

Left factoring rewrites A→αβ|αγ using:
A→αA′
A′→β|γ
It postpones a choice until after the common prefix. It does not by itself remove ambiguity or guarantee LL(1). LR parsers can handle suitable left-recursive grammars; ordinary unmodified recursive descent can recurse forever before consuming input.

Question sources

Compiler Design review questions · Question 34

Question 182 | NS0262

In attribute grammar terminology, which of the following is an 'inherited' attribute?

A. The value of an expression node computed from its children |

B. The type environment passed from a function declaration down to its body

C. The size of an array determined from its element type and length |

D. The result type of an arithmetic operation

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer B: A type environment passed from a declaration into its body is inherited.

COMP24 explicitly gives the declaration’s type environment passed into its body as an inherited attribute. This carries context inward rather than computing a parent from child values. Expression values, array sizes computed from components and arithmetic result types are ordinarily synthesized examples when obtained from the node’s own child information.

Why each choice is right or wrong

A - Incorrect: Computing an expression from its children is synthesized upward flow.

B - Correct: The surrounding declaration supplies type context to the nested body, matching the explicit inherited example.

C - Incorrect: Computing size from element information and length is a synthesized result, not supplied surrounding context.

D - Incorrect: An operation’s result type derived from operands is normally synthesized from its child information.

Primary note proof

Study-note section COMP24

A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions. An annotated parse tree displays computed attribute values; it is not itself an

inherited or synthesized attribute. Synthesized attributes flow from children to a parent; inherited attributes carry context from parent or siblings.

Study-note section COMP24

For E→E1+T, E.val=E1.val+T.val synthesizes an expression value. A declaration’s type environment passed into its body is inherited. S-attributed definitions use only synthesized attributes and fit LR reduction-time evaluation naturally; suitable LL parsers can also compute them. L-attributed definitions restrict dependencies so a left-to-right traversal can evaluate inherited context. Attribute dependency cycles can prevent evaluation.

Question sources

Compiler Design review questions · Question 35

Question 183 | NS0263

A compiler emits the error 'variable used before initialization'. This is detected during:

A. Lexical analysis |

B. Syntax analysis

C. Semantic analysis / data flow analysis |

D. Code generation

Hint

Track which facts can hold along one path and which must hold along every path. Apply the specified direction, transfer rule and join operation.

Answer C: Semantic/data-flow analysis.

COMP29 distinguishes definite assignment, which asks whether initialization occurs on every incoming path, from reaching definitions alone. Such control/data-flow facts go beyond recognizing legal tokens or grammar. The error therefore belongs to semantic/data-flow analysis before target emission, even though exact acceptance and diagnostic rules depend on the language.

Why each choice is right or wrong

A - Incorrect: Scanning recognizes identifier tokens but does not prove initialization along every control-flow path.

B - Incorrect: A grammatically valid variable use can still occur before initialization.

C - Correct: Semantic data-flow analysis checks that required initialization precedes each use on every relevant path.

D - Incorrect: Target generation ordinarily consumes checked input rather than supplying the primary definite-assignment analysis.

Primary note proof

Study-note section COMP29

> Liveness is backward may-analysis: a value might be read before redefinition on some future path. OUT[B]=union of IN over successors; IN[B]=USE[B]∪(OUT[B]−DEF[B]). Iterate to a fixed point. Liveness supports register allocation and dead-value removal. Definite-assignment analysis asks whether initialization occurs on every path before use; reaching definitions alone does not prove this.

Question sources

Compiler Design review questions · Question 36

Question 184 | NS0264

In graph-coloring register allocation, what is an 'interference graph'?

A. A graph where nodes represent machine instructions and edges represent data dependencies |

B. A graph where nodes represent variables and edges connect pairs of variables that are simultaneously live

C. A graph where nodes represent registers and edges represent compatibility |

D. A graph representing the control flow of the program

Hint

Match stored information to its lifetime, access path and calling-convention role; distinguish compile-time organization from run-time storage.

Answer B: A graph connecting values that must coexist in distinct registers.

COMP36 models concurrently needed values as vertices joined by interference edges, with physical registers represented by colors. Thus the variable/live-range graph in B is the intended model. Instruction dependencies and control-flow edges have different meanings; a graph of register compatibility reverses the allocation model's vertices and does not express live-value interference.

Why each choice is right or wrong

A - Incorrect: Instruction-dependency edges constrain scheduling rather than describing interference among live values.

B - Correct: Variables or live ranges form vertices; an edge requires distinct registers for simultaneously needed values.

C - Incorrect: Registers are colors in this model, not the vertices whose live ranges interfere.

D - Incorrect: A CFG connects basic blocks by possible transfers, not values by register interference.

Primary note proof

Study-note section COMP36

> A value is live where it can be used before redefinition. In the interference model, values that must coexist in distinct registers are connected by an edge; each register is a color. Nonoverlapping live ranges may share a register. If a valid assignment cannot fit the available registers, spilling places selected values in memory and inserts loads/stores.

Study-note section COMP36

> The general graph k-colorability decision problem is NP-complete for fixed k≥3; optimal register assignment/spill minimization is a hard optimization problem in general, so practical allocators use heuristics.

Question sources

Compiler Design review questions · Question 37

Question 185 | NS0265

What is 'dead code' in the context of compiler optimization?

A. Code that causes a runtime error |

B. Code that is syntactically incorrect

C. Code that is unreachable or whose computed results are never used |

D. Code that depends on uninitialized variables

Hint

Trace whether computation is evaluated, moved, shared or removed. Preserve observable effects and use the stated arithmetic and target assumptions.

Answer C: Unreachable code or unused computations without required effects.

COMP30 permits removing unreachable code and unused computations only when evaluation has no necessary side effects. C states the intended two dead-code categories, with this semantic caveat. Runtime faults, syntax defects and uninitialized uses are different error categories, not definitions of code being safely removable. An unused volatile read or effectful call may still be required.

Why each choice is right or wrong

A - Incorrect: A runtime fault does not make arbitrary reachable code safely removable as dead code.

B - Incorrect: Syntactically invalid code is a compilation error, not the definition of dead computation.

C - Correct: Unreachability or unused results identify candidates, provided necessary observable side effects are preserved.

D - Incorrect: An uninitialized dependency is an analysis or correctness issue, not automatically dead code.

Primary note proof

Study-note section COMP30

> Constant folding evaluates constant expressions during compilation: 3*4→12. Constant propagation substitutes known values: x=3; y=x+2 becomes y=3+2 before folding. Copy propagation substitutes a copied name while its source remains unchanged. Dead-code elimination removes unreachable code or computations whose values are unused and whose evaluation has no necessary side effects.

Question sources

Compiler Design review questions · Question 38

Question 186 | NS0266

What is the precise relationship between the general term compiler and the term transpiler?

A. A transpiler is a source-to-source compiler; compilers can also target assembly, machine code or bytecode.

B. A compiler is faster than a transpiler

C. A transpiler performs more optimizations than a compiler |

D. There is no meaningful distinction

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: A transpiler is a source-to-source compiler; compilers can also target assembly, machine code or bytecode.

A compiler translates a source program to an equivalent target representation, which may be machine code, assembly, bytecode or another source language. A transpiler is the source-to-source subtype of compiler. This gives a meaningful target-based distinction while retaining the shared category. Speed and optimization count depend on implementations and workloads; neither defines the relationship between the two terms.

Why each choice is right or wrong

A - Correct: A transpiler is a compiler whose target is another source language.

B - Incorrect: There is no universal speed ordering.

C - Incorrect: Optimization count does not define the subtype.

D - Incorrect: The source-to-source target is a meaningful distinction.

Primary note proof

Study-note section COMP01

> A compiler translates a source program into an equivalent target representation while preserving defined observable behavior. The target may be machine code, assembly, bytecode or another source language. A transpiler is a source-to-source compiler, not a fundamentally separate noncompiler technology. A compiler preserves language semantics, not the programmer's unstated intention.

Study-note section COMP01

> An interpreter executes a program representation; it need not work one source line at a time. Systems can compile to bytecode and then interpret or JIT-compile it. AOT compilation happens before execution; JIT compilation happens during execution and may use runtime profiles. Neither is universally faster; startup cost, optimization and workload matter.

Question sources

Compiler Design review questions · Question 39

Question 187 | NS0267

In the context of lexical analysis, 'lexeme' vs 'token' means:

A. Lexeme is the abstract category; token is the actual text |

B. Lexeme is the actual sequence of characters in the source; token is the abstract category with optional attribute value

C. They are synonymous terms |

D. Lexeme refers to keywords only; token refers to all other units

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: Lexeme is the source text; token is its category with optional attributes.

COMP04 directly distinguishes the actual character sequence from its token category and optional attribute. For example, total is a lexeme that may produce an ID token associated with a symbol entry. Neither term is restricted to keywords; the same distinction applies to identifiers, numeric literals, operators and other lexical units.

Why each choice is right or wrong

A - Incorrect: This reverses the roles: the lexeme is concrete text and the token identifies its class.

B - Correct: The concrete source sequence is the lexeme; category and optional information form the token.

C - Incorrect: The note explicitly separates the two abstractions rather than treating them as synonyms.

D - Incorrect: Lexemes include identifiers, literals and operators, not just keywords.

Primary note proof

Study-note section COMP04

> A token is a category with optional attribute, such as (ID, symbol-entry) or (INT_LITERAL,42). A lexeme is the actual character sequence, such as total or 42. A pattern specifies which lexemes belong to a token class. The scanner emits an ordered token stream, not an unordered mathematical set.

Study-note section COMP04

> Keywords, identifiers, literals, operators and punctuation are token classes. Whitespace and comments are usually skipped while line/column positions are retained. In indentation-sensitive or newline-sensitive languages, layout can carry tokens. Parse-tree construction is a parser task; scanning can intern identifier spellings, but declaration scope and types need later analysis.

Question sources

Compiler Design review questions · Question 40

Question 188 | NS0268

Under the general interference-graph model with fixed k≥3, which task is an NP-complete decision problem?

A. Converting an NFA to a DFA |

B. Constructing a LL(1) parsing table

C. Deciding whether an arbitrary interference graph can use k registers without spilling, for fixed k≥3

D. Computing FIRST sets for a CFG

Hint

Match stored information to its lifetime, access path and calling-convention role; distinguish compile-time organization from run-time storage.

Answer C: Deciding whether an arbitrary interference graph can use k registers without spilling, for fixed k≥3

In the general interference model, simultaneously needed values are adjacent and registers are colors. Deciding whether the graph can be colored with k registers without spilling is general graph k-colorability, NP-complete for fixed k≥3. Optimal allocation is instead an optimization task. The other alternatives are construction or fixed-point tasks and do not state this membership-plus-hardness decision problem.

Why each choice is right or wrong

A - Incorrect: NFA determinization is a construction task, even when its output is exponential.

B - Incorrect: LL(1) table construction is not the specified graph-coloring decision task.

C - Correct: No-spill allocation is k-colorability under the stipulated general model.

D - Incorrect: FIRST-set computation is a fixed-point task rather than this NP-complete decision form.

Primary note proof

Study-note section COMP36

> A value is live where it can be used before redefinition. In the interference model, values that must coexist in distinct registers are connected by an edge; each register is a color. Nonoverlapping live ranges may share a register. If a valid assignment cannot fit the available registers, spilling places selected values in memory and inserts loads/stores.

Study-note section COMP36

> The general graph k-colorability decision problem is NP-complete for fixed k≥3; optimal register assignment/spill minimization is a hard optimization problem in general, so practical allocators use heuristics.

Question sources

Compiler Design review questions · Question 41

Question 189 | NS0269

What is the 'reaching definitions' data flow problem used for?

A. Determining which variables are live at each program point |

B. Determining which variable assignments might reach a given program point without being overwritten

C. Determining which expressions are available at each program point |

D. Determining which functions can be called from a given call site

Hint

Track which facts can hold along one path and which must hold along every path. Apply the specified direction, transfer rule and join operation.

Answer B: Assignments that may reach a point without being overwritten.

COMP29 directly defines reaching definitions as assignments that can arrive at a point without a replacement definition on the path. Its join is union because one possible path suffices. Liveness asks about future uses, available expressions asks about previous computations on every path, and possible call targets is a separate analysis question.

Why each choice is right or wrong

A - Incorrect: Future reads before redefinition are the liveness question, not reaching definitions.

B - Correct: This is the exact assignment-and-no-overwrite property defined for reaching definitions.

C - Incorrect: Available expressions concerns prior computations with unchanged operands, not assignment definitions.

D - Incorrect: Possible function targets concern call-graph or points-to analysis rather than reaching assignment definitions.

Primary note proof

Study-note section COMP29

> Reaching definitions is forward may-analysis: which assignments can reach this point without a replacement definition? OUT=GEN∪(IN−KILL), with IN the union of predecessor OUT sets. Available expressions is usually forward must-analysis: an expression has been computed on every relevant path and its operands have not changed. Join with intersection.

Question sources

Compiler Design review questions · Question 42

Question 190 | NS0270

In predictive (LL(1)) parsing, when should the parser use an ε-production A → ε?

A. When the lookahead token is in FIRST(A) |

B. When the lookahead token is in FOLLOW(A) and no other production for A begins with that token

C. When the stack is empty |

D. Whenever the next token is not a keyword

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: Use ε on an appropriate FOLLOW(A) lookahead with no competing alternative.

COMP14 enters a nullable alternative under the left-hand nonterminal's FOLLOW symbols and requires no conflicting production in that cell. An epsilon reduction consumes no input. FIRST(A) alone can refer to a nonempty alternative, while stack emptiness and token being a keyword do not define nullable-alternative selection. Thus B supplies the correct lookahead rule.

Why each choice is right or wrong

A - Incorrect: A FIRST terminal may require a nonempty alternative; FIRST(A) alone does not select epsilon.

B - Correct: FOLLOW lookahead selects the nullable alternative when its predictive table cell has no competing production.

C - Incorrect: An empty stack is not the rule for selecting a production for a pending nonterminal.

D - Incorrect: Keyword status has no general relation to a nonterminal's nullable-production table entries.

Primary note proof

Study-note section COMP14

> For every A→α, put it in M[A,a] for each terminal a in FIRST(α). If α is nullable, also enter it under every symbol in FOLLOW(A). More than one production in a cell is a conflict. Pairwise SELECT sets must be disjoint; SELECT(α)=FIRST(α)−{ε}, plus FOLLOW(A) when α is nullable. This also detects two nullable alternatives.

Study-note section COMP14

> Initialize stack with $S. Match and pop terminals; for a nonterminal consult its row and lookahead, then push the chosen RHS in reverse. An ε production consumes no input. Accept when stack and input reach $. For A→a|ε with b∈FOLLOW(A), choose a on lookahead a and ε on b.

Question sources

Compiler Design review questions · Question 43

Question 191 | NS0271

Which of the following correctly describes a 'dominator' in a control flow graph?

A. A node that is executed before all other nodes unconditionally |

B. Node D dominates node N if every path from the entry node to N must pass through D

C. The node with the most outgoing edges |

D. The unique successor of every conditional branch

Hint

Identify what the representation stores, what it omits and which later transformations it must support; separate a representation property from an algorithmic task.

Answer B: Every entry-to-N path passes through D.

COMP28 defines domination relative to a particular node N: every path from entry to N must pass D. This is not a claim that D precedes every node in the program, nor is domination measured by out-degree or being a unique successor. The definition concerns unavoidable traversal on paths reaching the specified node.

Why each choice is right or wrong

A - Incorrect: A dominator is defined relative to a node, not necessarily as preceding all program nodes.

B - Correct: This exactly states the all-entry-to-N-paths definition in COMP28.

C - Incorrect: The number of outgoing edges does not establish unavoidable traversal on paths.

D - Incorrect: Branch successor uniqueness is unrelated to the general dominator definition.

Primary note proof

Study-note section COMP28

> A basic block is a maximal straight-line instruction sequence entered at its beginning and exited at its end. Leaders are the first instruction, every jump target, and each instruction after a branch. Start a block at each leader and continue until the next leader.

Study-note section COMP28

> A control-flow graph has blocks as vertices and possible next-block transfers as edges, including fall-through. A node D dominates N when every entry-to-N path passes through D. For an edge N→H where H dominates N, H is a loop header of a natural loop. This structure supports loop analysis. An expression DAG records value dependencies, not arbitrary control flow.

Question sources

Compiler Design review questions · Question 44

Question 192 | NS0272

In SLR(1) parsing, the reduce action for production A → α is triggered when the lookahead is in:

A. FIRST(A) |

B. FOLLOW(A)

C. FIRST(α) |

D. FOLLOW(α)

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: FOLLOW(A), once the item for A→α is complete.

COMP17 says SLR uses LR(0) states and FOLLOW(A) for reduction by a completed A production. The completed item establishes that the right side has been recognized; FOLLOW restricts which lookaheads permit reducing it. FIRST information predicts beginnings instead, and FOLLOW of an arbitrary right-hand string is not the table rule used here.

Why each choice is right or wrong

A - Incorrect: FIRST(A) concerns beginnings of derived strings rather than lookaheads after a completed A.

B - Correct: SLR places a completed A-production reduction under the lookaheads in FOLLOW(A).

C - Incorrect: FIRST of the right side predicts starting tokens, not permitted post-recognition reduction lookaheads.

D - Incorrect: SLR uses FOLLOW of the left-hand nonterminal A, not this right-side expression.

Primary note proof

Study-note section COMP17

> SLR(1) uses LR(0) states and FOLLOW(A) for reduction by a completed A production. Canonical LR(1) attaches context-specific lookahead to items. LALR(1) merges canonical states with the same LR(0) core, combining lookaheads; merging can introduce reduce/reduce conflicts. The standard grammar-class hierarchy is LR(0)⊆SLR(1)⊆LALR(1)⊆LR(1), with increasing capability and often table size.

Study-note section COMP17

> A shift/reduce conflict offers both consuming a token and reducing; reduce/reduce offers two reductions. An SLR or LALR conflict does not by itself prove the grammar ambiguous. Canonical LR(1) handles LR(1) grammars, not every unambiguous CFG. Precedence and associativity can resolve suitable expression conflicts; changing parser names alone does not repair all ambiguity.

Question sources

Compiler Design review questions · Question 45

Question 193 | NS0273

What is 'escape analysis' and how does it help optimization?

A. Analysis to detect out-of-bounds array accesses; helps prevent security vulnerabilities |

B. Analysis to determine whether an object's reference can 'escape' the function where it is created; enables stack allocation of non-escaping objects

C. Analysis to find unreachable exception handlers; enables dead code elimination |

D. Analysis to track pointer arithmetic; enables bounds checking elimination

Hint

Match stored information to its lifetime, access path and calling-convention role; distinguish compile-time organization from run-time storage.

Answer B: Determine whether references outlive or escape a region, enabling safe allocation changes.

COMP37 asks whether an object/reference can outlive or be accessed outside its allocating region. A proven nonescaping object may use stack storage or be eliminated when lifetime and identity are preserved. Bounds checking, exception reachability and pointer arithmetic are separate analyses and do not define the reference-escape property described.

Why each choice is right or wrong

A - Incorrect: Array bounds concern index validity rather than whether references leave their allocation region.

B - Correct: Reference escape information can justify stack allocation when lifetime and identity remain correct.

C - Incorrect: Unreachable exception handlers concern control-flow reachability, not reference lifetime or external access.

D - Incorrect: Pointer arithmetic tracking does not by itself define escape beyond a function or allocation region.

Primary note proof

Study-note section COMP37

> Static storage lasts throughout execution; stack frames support nested calls and recursion; heap storage supports lifetimes beyond a call. A frame may contain parameters, return address, saved registers, locals and temporaries. Sizes, alignment and offsets are target-dependent; a two-byte int is an example, not a C++ guarantee.

Study-note section COMP37

> A calling convention specifies argument and result locations, caller/callee-saved registers, stack alignment and frame responsibilities. Escape analysis asks whether an object/reference can outlive or be accessed outside a region such as its allocating call. Proven nonescaping allocations may use stack storage or be eliminated by scalar replacement, provided identity and lifetime semantics remain correct.

Question sources

Compiler Design review questions · Question 46

Question 194 | NS0274

Which of the following is a valid reason why LALR(1) parsers may have reduce/reduce conflicts that canonical LR(1) parsers do not?

A. LALR(1) uses a larger parsing table than canonical LR(1) |

B. LALR(1) merges LR(1) states with the same core but different lookaheads, potentially merging incompatible lookahead sets

C. LALR(1) does not support left-recursive grammars |

D. LALR(1) computes FOLLOW sets incorrectly

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: Merging equal-core LR(1) states can combine incompatible reduction lookaheads.

COMP17 directly explains that LALR merges canonical states with identical LR(0) cores while unioning their lookahead sets. Distinctions that prevented competing reductions in separate states can disappear, creating reduce/reduce conflicts. This is not due to incorrect FOLLOW computation or rejection of left recursion; the loss comes from deliberate state merging.

Why each choice is right or wrong

A - Incorrect: Merging aims to reduce state count, and a larger table is not the reason for new conflicts.

B - Correct: Combining lookaheads of equal-core states can make previously separated reductions compete.

C - Incorrect: LALR can handle suitable left-recursive grammars, as other LR methods can.

D - Incorrect: The conflict can arise from correct merging, without any erroneous FOLLOW-set computation.

Primary note proof

Study-note section COMP17

> SLR(1) uses LR(0) states and FOLLOW(A) for reduction by a completed A production. Canonical LR(1) attaches context-specific lookahead to items. LALR(1) merges canonical states with the same LR(0) core, combining lookaheads; merging can introduce reduce/reduce conflicts. The standard grammar-class hierarchy is LR(0)⊆SLR(1)⊆LALR(1)⊆LR(1), with increasing capability and often table size.

Study-note section COMP17

A shift/reduce conflict offers both consuming a token and reducing; reduce/reduce offers two reductions. An SLR or LALR conflict does not by itself prove the grammar ambiguous. Canonical LR(1) handles LR(1) grammars, not every unambiguous CFG. Precedence and associativity can resolve suitable expression conflicts; changing parser names alone does not repair all ambiguity.

Question sources

Compiler Design review questions · Question 47

Question 195 | NS0275

Which is a common implementation mechanism for runtime virtual dispatch in an object-oriented language, when the receiver's class cannot be resolved statically?

A. Static binding of function calls at compile time |

B. A virtual method table (vtable) per class, with function pointers resolved at runtime

C. Inline expansion of all polymorphic calls |

D. Name mangling to distinguish overloaded functions

Hint

Match stored information to its lifetime, access path and calling-convention role; distinguish compile-time organization from run-time storage.

Answer B: A virtual method table (vtable) per class, with function pointers resolved at runtime

A common implementation stores virtual-method entries in a per-class vtable and lets an object refer to the appropriate table. A virtual call selects the entry using the receiver's runtime class. Static binding alone does not provide that selection for an unknown class. Mandatory inlining is not generally possible, and name mangling distinguishes linker symbols rather than selecting overrides at runtime.

Why each choice is right or wrong

A - Incorrect: Static binding alone does not choose by an unknown runtime class.

B - Correct: A per-class vtable is the common mechanism described in the note.

C - Incorrect: Unknown runtime receivers prevent mandatory inlining of all polymorphic calls.

D - Incorrect: Name mangling identifies symbols rather than selecting overrides.

Primary note proof

Study-note section COMP38

Inlining replaces a call with a suitably renamed copy of its body, preserving argument evaluation, returns and side effects. It can remove call overhead and expose optimization opportunities, but may increase code size and instruction-cache pressure. A recursive function cannot simply be expanded without limit.

Study-note section COMP38

Virtual dispatch chooses an overridden method based on a receiver's runtime class. A common implementation uses a per-class virtual-method table and an object pointer to the table; the call loads the selected entry. This is an implementation strategy, not a universal language requirement. Name mangling distinguishes linker symbols, often for overloads; it is not itself runtime method selection. Proven receiver types may permit devirtualization.

Question sources

Compiler Design review questions · Question 48

Question 196 | NS0276

Which of the following is true about a 'well-formed' grammar for LL(1) parsing?

A. It may contain left recursion |

B. For every non-terminal A with productions A → α1 | α2 | ... | αn, no two alternatives share a common token in their FIRST sets, and if any αi derives ε, then FIRST(αj) and FOLLOW(A) are disjoint for all j ≠ i

C. It must be ambiguous |

D. It must have at least one ε-production

Hint

Use construction direction, lookahead and grammar restrictions. Distinguish algorithm families, implementations and coverage before comparing them.

Answer B: Disjoint predictive selections, including nullable FIRST/FOLLOW constraints.

COMP14 requires that alternatives for a nonterminal select disjoint lookahead sets. B gives the standard FIRST separation and nullable-alternative FOLLOW restriction; shared epsilon alternatives would also violate pairwise FIRST disjointness. Left recursion and useful ambiguity are not acceptable LL(1) properties, and epsilon productions are optional rather than required.

Why each choice is right or wrong

A - Incorrect: Ordinary LL(1) grammars cannot retain left recursion that recurs without input consumption.

B - Correct: Disjoint alternative selections prevent multiple productions from occupying the same predictive table cell.

C - Incorrect: A useful ambiguous grammar cannot give a unique deterministic LL(1) parse.

D - Incorrect: A grammar may be LL(1) without any epsilon production.

Primary note proof

Study-note section COMP14

For every A→α, put it in M[A,a] for each terminal a in FIRST(α). If α is nullable, also enter it under every symbol in FOLLOW(A). More than one production in a cell is a conflict. Pairwise SELECT

sets must be disjoint; SELECT(α)=FIRST(α)−{ε}, plus FOLLOW(A) when α is nullable. This also detects two nullable alternatives.

Study-note section COMP14

Initialize stack with $S. Match and pop terminals; for a nonterminal consult its row and lookahead, then push the chosen RHS in reverse. An ε production consumes no input. Accept when stack and input reach $. For A→a|ε with b∈FOLLOW(A), choose a on lookahead a and ε on b.

Question sources

Compiler Design review questions · Question 49

Question 197 | NS0277

The primary difference between a 'compiler' and a 'just-in-time (JIT) compiler' is:

A. A JIT compiler produces faster code than an ahead-of-time compiler always |

B. A JIT compiler compiles code at runtime (during program execution), allowing it to use runtime profiling information for optimization, while an AOT compiler compiles before execution

C. A JIT compiler does not perform any optimization |

D. A JIT compiler only works for interpreted languages

Hint

Identify each activity’s input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer B: JIT compiles during execution; AOT compiles before execution.

COMP01 defines AOT before execution and JIT during execution, possibly using runtime profiles. Both are compilation, so the heading compiler versus JIT is better understood as AOT versus JIT. The timing contrast in B is correct, while no universal speed guarantee, absence of optimization or restriction to interpreted languages follows.

Why each choice is right or wrong

A - Incorrect: Neither AOT nor JIT is universally faster; startup and workload affect the comparison.

B - Correct: Runtime compilation can use execution profiles, whereas AOT performs compilation before the run.

C - Incorrect: JIT may optimize using runtime information rather than performing no optimization.

D - Incorrect: JIT is an execution-time compilation strategy, not a technique restricted to a separately defined interpreted-language category.

Primary note proof

Study-note section COMP01

A compiler translates a source program into an equivalent target representation while preserving defined observable behavior. The target may be machine code, assembly, bytecode or another source language. A transpiler is a source-to-source compiler, not a fundamentally separate noncompiler technology. A compiler preserves language semantics, not the programmer's unstated intention.

Study-note section COMP01

An interpreter executes a program representation; it need not work one source line at a time. Systems can compile to bytecode and then interpret or JIT-compile it. AOT compilation happens before execution; JIT compilation happens during execution and may use runtime profiles. Neither is universally faster; startup cost, optimization and workload matter.

Question sources

Compiler Design review questions · Question 50

Question 198 | NS0299

What is the main purpose of syntax-directed translation in a compiler?

A. To interleave semantic analysis with syntax analysis using attributes attached to grammar symbols

B. To generate optimized machine code directly

C. To tokenize the source code

D. To replace the parser with a finite automaton

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer A: Associate semantic computations with grammatical structure and attributes.

COMP24 attaches attributes and semantic rules to grammar symbols and productions; COMP25 executes translation actions during processing. This permits semantic work to follow parsing structure, as A states. It does not inherently bypass IR to optimized machine code, reduce to tokenization, or replace the parser with a finite automaton.

Why each choice is right or wrong

A - Correct: Attributes and rules connect grammatical structure to semantic computations during translation.

B - Incorrect: Direct optimized machine-code production is not the required or exclusive purpose of syntax-directed translation.

C - Incorrect: Tokenization is lexical analysis rather than grammar-directed semantic computation.

D - Incorrect: Finite automata recognize ordinary token patterns and do not replace general nested-syntax parsing.

Primary note proof

Study-note section COMP24

> A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions.

Study-note section COMP25

> A syntax-directed translation scheme embeds executable actions in productions; the actions run while translating a source program, not while generating the compiler. Attributes may compute types, values, AST pointers or generated code. The scheme defines an input-to-output translation, not merely an input language.

Question sources

Computer Science Exit Exam · 2018 · Question 16

Question 199 | NS0303

Straight directed translation uses:

A. Purely lexical rules

B. Backtracking algorithms

C. Grammar without attributes

D. Grammar with sematic rule

Hint

Track where each computed value comes from and when each production action runs. Distinguish the representation from its attached information.

Answer D: Grammar with semantic rules, reading the misspelled term as syntax-directed translation.

The stem's Straight directed translation appears to be a damaged rendering of syntax-directed translation. Under that explicit reading, COMP24 defines a CFG with attributes and semantic rules, and COMP25 embeds translation actions in productions. D expresses that structure despite its spelling error. The other choices omit semantics or prescribe unrelated implementation details.

Why each choice is right or wrong

A - Incorrect: Pure lexical rules do not supply grammar-directed semantic attributes and computations.

B - Incorrect: Backtracking is a parsing implementation choice, not the defining translation structure.

C - Incorrect: A grammar without semantic attributes or rules does not describe the syntax-directed definition requested.

D - Correct: Reading sematic as semantic, this is the grammar-plus-semantic-rules structure of syntax-directed translation.

Primary note proof

Study-note section COMP24

> A syntax-directed definition is a CFG with attributes on grammar symbols and semantic rules on productions.

Study-note section COMP25

> A syntax-directed translation scheme embeds executable actions in productions; the actions run while translating a source program, not while generating the compiler. Attributes may compute types, values, AST pointers or generated code. The scheme defines an input-to-output translation, not merely an input language.

Question sources

Computer Science Exit Exam · 2018 · Question 21

Question 200 | NS0331

A compiler has produced checked three-address code for a function. Which description incorrectly places the ensuing target-lowering task within the standard compilation pipeline?

A. Code generation can be considered as the start phase of compilation

B. The target program is the output of the code generator

C. Checked intermediate code is supplied as input to the code generator.

D. It used to produce the target code for three-address statements.

Hint

Identify each activity's input and output. Distinguish character recognition, grammatical structure, semantic checks and target implementation.

Answer A: Code generation can be considered as the start phase of compilation

Code generation uses checked intermediate code to produce target instructions after the frontend has analyzed the source. It can lower each three-address operation into a suitable target sequence. Lexical analysis, not code generation, begins the standard pipeline. The input condition here means checked IR suitable for lowering; it is not a claim that all possible program bugs have been proved absent.

Why each choice is right or wrong

A - Correct: Target lowering follows scanning, parsing, semantic checks and IR construction; it does not start compilation.

B - Incorrect: The generator's output is target code.

C - Incorrect: Checked IR is the stated input to target lowering.

D - Incorrect: Three-address statements can be translated into target instructions.

Primary note proof

Study-note section COMP03

> The conventional sequence is lexical analysis → syntax analysis → semantic analysis → intermediate-code generation → optimization → target-code generation. Scanning turns characters into tokens; parsing builds structure; semantics checks declarations and types; synthesis produces implementation code. A phase is a logical activity; a pass is a traversal that can combine activities.

Study-note section COMP34

> Code generation lowers checked IR to a target form.

Question sources

Computer Science Exit Exam · 2018 · Question 58